

Programátorská a administrátorská dokumentace k projektu

MONEYMAKER



vedoucí projektu
RNDr. David Hoksza

David Matoušek

Jan Raszyk

Petr Švec

Martina Tomisová

2008/2009

UNIVERZITA KARLOVA V PRAZE

matematicko-fyzikální fakulta



Projekt *MoneyMaker* byl vytvořen
v rámci předmětu *Softwarový projekt (NPRG023)*.

Vedoucí projektu:

RNDr. David Hoksza

Autoři projektu:

David Matoušek

Jan Raszyk

Petr Švec

Martina Tomisová

Obsah

Obsah.....	5
Úvod.....	8
Funkční popis.....	8
Jak se píše plugin.....	10
Požadavky.....	11
Požadavky na software.....	11
Pro server.....	11
Pro uživatelský počítač.....	11
Požadavky na hardware.....	11
Pro server.....	11
Pro uživatelský počítač.....	11
Programátorská a administrátorská dokumentace.....	12
Cribber.....	12
MoneyDownload.....	12
Jak program pracuje.....	14
SportsDownload.....	15
Jak program pracuje.....	17
BusinessDemo a Businessman.....	20
Businessman.....	21
MoneyPack.....	21
SportPack.....	23
Core.....	31
PluginBase.....	31
BusinessDemo.....	31
Co je úkolem BusinessDemo.....	31
Princip práce BusinessDemo.....	32
Formát datových souborů.....	32
Vytváření datových souborů.....	34
Pro sporty.....	34
Businessman.....	34
Jak Businessman pracuje.....	34
Struktura pluginu.....	36
Třída PluginBase.....	36
Moderátor.....	37
Princip práce Moderátora.....	38
Jednotlivé komponenty Moderátora.....	38
Konfigurační soubor.....	38
Autentizace uživatele.....	39
Validace vstupu.....	39
Filtr dat.....	41
Lokalizace.....	42
Typy.....	42
Práce s databází.....	42

Hlavní formulář aplikace.....	44
Genesis.....	46
Princip práce programu.....	47
Druhy proměnných vzhledem k možnostem křížení.....	48
Konfigurační soubor.....	48
Logy.....	49
Occupant.....	49
Obecná struktura průběhu načítání stránky.....	49
Některé části podrobněji.....	50
Třída Page.....	50
Lokalizace.....	50
Tabulky.....	51
Formuláře.....	52
Relace.....	53
Práce s databází.....	53
Použité veřejně dostupné moduly: PEAR a jQuery.....	54
Další užitečné funkce.....	54
Killguard.....	55
Ohodnocování instancí.....	55
Spočtení rozdílu mezi skutečností a odhadem.....	56
Ukončování instancí.....	57
Křížení instancí.....	57
Genetický inženýr.....	58
Matematické a statistické knihovny.....	58
ILNumerics.....	59
Statistická knihovna.....	59
DnAnalytics.....	59
Databáze.....	59
Rejstřík databázových tabulek.....	59
Uživatelé.....	60
T101Users.....	60
T102UserAdditionInformation.....	61
T103Rights.....	63
Souhrn.....	65
Systémová data a data strategií.....	65
T105SystemDials.....	65
T106DataDials.....	67
T110SystemLogs.....	68
Souhrn.....	69
Kurzy měn.....	69
T201Money.....	69
T202MoneyHistory.....	71
T205MoneyBets.....	72
Souhrn.....	73
Data o strategiích a jejich instancích.....	74
T301InvestStrategy.....	74
T302InvestStrategyVariableRange.....	76
T303InvestInstance.....	77
T304InvestInstanceVariables.....	79
T305InvestInstanceSerializedVariables.....	80
T310InvestInstanceLog.....	81

T314InvestInstanceVariablesHistory.....	83
Souhrn.....	84
Sportovní data.....	84
T401SuperSports.....	84
T402Sports.....	85
T403Competitions.....	87
T404CompetitionYears.....	88
T405Matches.....	90
T406Opportunities.....	91
T410Teams.....	93
T420OpportunitiesBet.....	95
Souhrn.....	97
Další funkce.....	98
T400InsertedMatchId.....	98
T400InsertedMatchScore.....	99
Schéma databáze jako celku.....	101
Instalace.....	102
Deinstalace.....	114
Údržba systému.....	115
Umístění logů.....	115
Konfigurační soubory.....	115
Co dělat když.....	115
Nestačí počet připojení k databázi.....	115
Bezpečnost.....	116

Úvod

Funkční popis

Systém *MoneyMaker* poskytuje uživateli možnost pohodlně si vyvíjet strategie pro investování do zahraničních měn a typování sportovních výsledků. Framework *MoneyMaker* strategii testuje a vylepšuje každý den na aktuálních datech a poskytuje uživateli výsledky.

Framework sleduje pouze méně likvidní trhy* (kurzy zahraničních měn). Cílem frameworku je pomoci uživateli vyvinout strategii, jež dobře predikuje za běžných okolností. Jakmile se ale ve světě děje něco mimořádného, co ovlivňuje hodnoty na trhu, tak se na ní spolehnout nedá.

Základní myšlenkou systému je křížení strategií. Každá strategie má vstupní parametry a na serveru běží v několika instancích, které se liší právě v hodnotě zadaných parametrů. Instance jedné strategie se mezi sebou každý den kříží genetickými algoritmy (kříží se pouze hodnoty parametrů). Instance jsou monitorovány, prodávající instance zabity a ty nejlépe vydělávající dále vzájemně kříženy. Na počátku si uživatel volí počáteční hodnoty parametrů a jejich smysluplný rozsah.

Uživatel, který chce framework využívat, si napíše plugin (libovolná .NET *assembly*), vyzkouší si ho pomocí jednoduchého programu založeného na historických datech (*BusinessDemo* aplikace). Až mu bude plugin fungovat, nahraje ho do frameworku, kde se bude pravidelně spouštět, vylepšovat a budou se vyhodnocovat jeho výsledky. Křížení parametrů strategií probíhá pouze ve frameworku, nikoli v *BusinessDemo*.

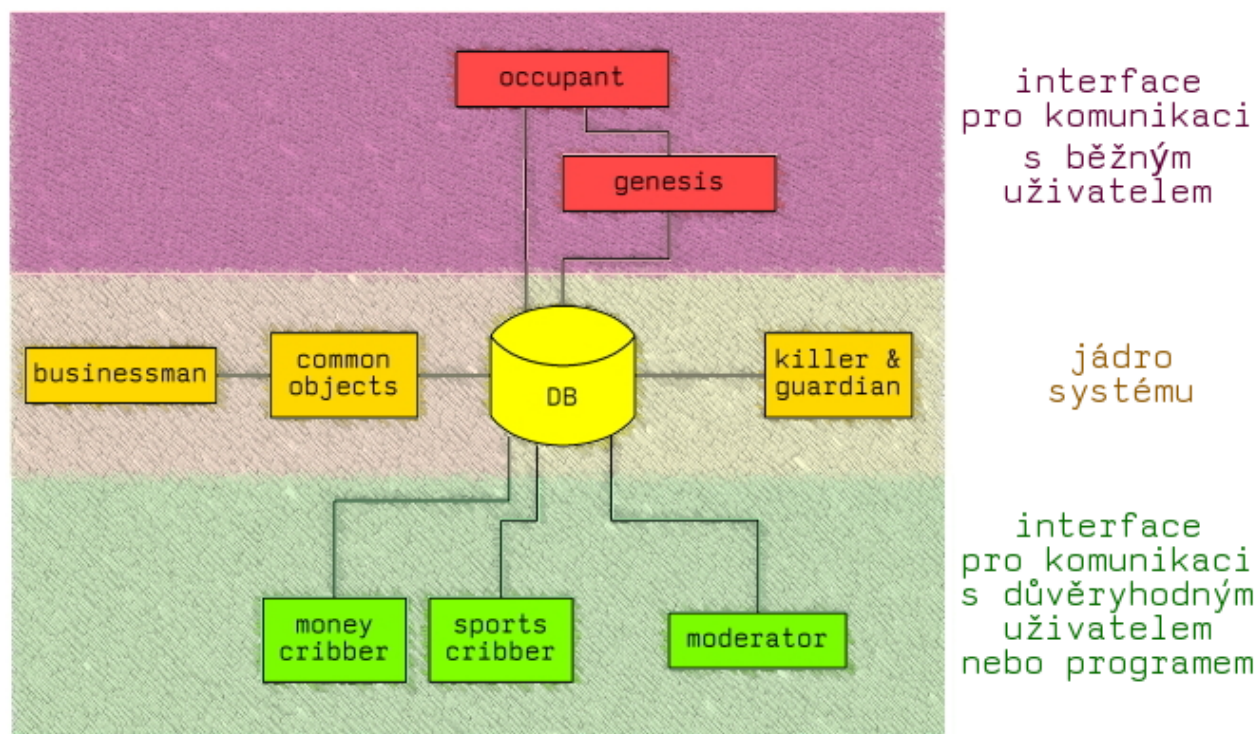
Celý framework běží v diskrétním čase (v cyklech po dnech). V každém cyklu

- se vyhodnotí úspěšnost tipů z předešlého dne,
- geneticky se zkříží nejlépe vydělávající strategie,
- každé živé instanci se aktualizují data (odpovídající aktuální situaci),
- získá se nový tip na další den od každé instance.

* *Likvidita představuje v obchodu a podnikání schopnost přeměnit aktiva do likvidní formy. To znamená schopnost rychle prodat nebo koupit určité komodity bez nebezpečí významné změny ceny. Likvidní trh se vyznačuje dostatečným množstvím kupujících a prodávajících. Dobrou charakteristikou likvidního trhu je schopnost uskutečnit další opakovaný obchod za stejnou cenu jako ten předchozí. Vysoce likvidní trh dokáže cenu komodity udržet i v případě nákupů a prodejů velkého množství. Ani velká nabídka nebo poptávka nedokáže cenu významně změnit. Různé trhy mají různý stupeň likvidity. Akciový trh má větší likviditu než trh s nemovitostmi. Největší likviditu mají hotové peníze. Převzato z <http://cs.wikipedia.org/wiki/Likvidita>.*

I když je strategie zpětně testována na historických datech (to dělají např. *BusinessDemo* aplikace), neví o tom. Framework se vždy tváří, jako by data z následujícího dne neznal, i když už je ve skutečnosti má k dispozici.

Hlavní část systému běží na serveru a uživatel do ní nahrává své strategie ze své klientské stanice. Tato hlavní část se skládá z několika samostatných částí, které spolu komunikují přes společné úložiště – databázi. Logická struktura projektu je znázorněna na obrázku 1.



Obrázek 1

Jádro systému je na obrázku znázorněno žlutě:

➔ **Businessman a Common Objects**

v každém cyklu (jednou denně) spouští všechny instance investičních strategií.

➔ **Killer & Guardian**

spouští se na začátku každého cyklu a ukončuje ty instance strategií, které investují špatně, stará se o genetické křížení investičních strategií, které investují dobře.

➔ **DB**

databáze, skladiště dat a zároveň komunikační kanál mezi jednotlivými částmi projektu.

Interface pro komunikaci s důvěryhodným uživatelem nebo programem je vyznačen zelenou barvou:

➔ **Cribbery**

starají se o stahování, konverzi a uložení dat do DB (aktualizují data týkající se kurzů měn, výsledků zápasů, vypsaných zápasů a jejich kurzů).

→ Moderátor

GUI serveru pro ovládání celého systému, je implementován jako tlustý klient .

Červeně je nakreslen **interface pro komunikaci s běžným uživatelem**:

→ Occupant

přístup pro uživatele systému přes webové rozhraní (tenký klient).

→ Genesis

Analyzuje plugin zadaný uživatelem v binární podobě za účelem zjištění použitých proměnných (a také zjišťuje, které se mají křížit a které ne).

Další součástí projektu jsou výše zmiňované programy *BusinessDemo* pro základní testování strategie před jejím nahráním do frameworku.

Jak se píše plugin

Uživatel si může napsat plugin v libovolném .NET jazyku. Je třeba vytvořit *dll* knihovnu, která obsahuje funkci (pojmenovanou `run`) implementující myšlenku strategie. Plugin má přes dané proměnné přístup k historickým datům, aby je mohl využívat při výpočtech a má k dispozici proměnnou, do které zapisuje svůj odhad na příští hodnotu. Uživatel má možnost označit některé proměnné pluginu jako proměnné, jejichž hodnoty budou přepsány při křížení. Více o psaní pluginu najdete v uživatelské dokumentaci.

Pro jednoduché otestování, zda je plugin dobře napsaný (zda se chová tak, jak uživatel chtěl), je třeba aplikace *BusinessDemo* a textový soubor obsahující historická data, která má plugin tipovat. Jednoduše se spustí *BusinessDemo* aplikace a jako parametry se zadají zmíněný textový soubor a soubor obsahující plugin. *BusinessDemo* vrátí uživateli informace o tom, jak si jeho plugin na datech vedl.

Potom uživatel nahraje plugin na server pomocí webového rozhraní, přes něj rovněž sleduje jeho výsledky.

Požadavky

Požadavky na software

Pro server

- Windows NT 5.0 ENG a novější, otestováno na Windows 2003 ENG
- .NET 3.5 a vyšší
- T-SQL MSSQL 2005 SP2 a vyšší
- Active Perl 5.10 a vyšší
- PHP 5 a vyšší
- Apache 2.2 a vyšší
- PEAR 1.7.1

Pro uživatelský počítač

- .NET 2.0 a vyšší
- webový prohlížeč

Požadavky na hardware

Pro server

- Připojení k Internetu s download rychlostí alespoň 256kb/s a upload rychlostí alespoň 64kb/s
- 1 GHz CPU nebo rychlejší
- 512MB RAM nebo více
- 2GB místa na disku

Pro uživatelský počítač

- Připojení k Internetu s rychlostí alespoň 64kb/s
- 450 MHz nebo rychlejší
- 192 MB RAM nebo více

Programátorská a administrátorská dokumentace

Použité jazyky:

C#, Delphi, T-SQL, PHP, Perl, JavaScript, XHTML, CSS, PascalScript

Cílová platforma:

Windows

Cribber

Jedná se o sadu dvou programů, které pravidelně stahují z webů data kurzů měn a sportovních výsledků a kurzů. Programy se jmenují MoneyDownload.exe a SportsDownload.exe.

MoneyDownload

Pravidelně stahuje z webu České národní banky na projektový server data kurzů měn (za zadané období).

Spouští ho *Task Scheduler*. Zdrojový kód je umístěn v adresáři cribber\MoneyDownload.

Konfigurační soubor pro tento program musí být umístěn ve stejném adresáři jako samotný spustitelný soubor. Ukázkový konfigurační soubor je umístěn v textu 1.

Text 1: konfigurační soubor pro MoneyDownload

```
<!-- Konfiguracni soubor Cribberu pro MoneyMaker -->

<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <appSettings>
    <!-- adresar pro ukladani logu -->
    <add key="LogDir" value="logs" />
  </appSettings>
</configuration>
```

```

    <!-- format jmena logovaciho souboru; musi obsahovat "{0}", to bude nahrazeno casovym
    razitkem (parametr DateTimeFormat) -->
    <add key="LogFile" value="log {0}.txt" />

    <!-- format casoveho razitka -->
    <add key="DateTimeFormat" value="yyyy-MM-dd hh_mm_ss" />

    <!-- HTTP adresa stranky, ze ktere se data nacistaji, za '{0}' bude dosazen pozadovany
    rok -->
    <add key="DownloadAddress" value="http://www.cnb.cz/cs/financni_trhy/devizovy_trh/kurz
    y_devizoveho_trhu/rok.txt?rok={0}" />

    <!-- pocet let zpět, za která chceme stáhnout data -->
    <add key="HistoryYears" value="1" />

    <!-- v pripade chyby bude odeslan informacni email -->

    <!-- adresa odesilatele -->
    <add key="MailFrom" value="soft.projekt@seznam.cz"/>

    <!-- adresa prijemce -->
    <add key="MailTo" value="petsvec@tiscali.cz"/>

    <!-- predmet emailu -->
    <add key="MailSubject" value="MoneyMaker - Cribber - money - ERROR"/>

    <!-- adresa serveru -->
    <add key="SmtpServer" value="192.168.80.2"/>

    <!-- pouzit Secure SMTP? hodnoty "false" nebo "true" -->
    <add key="SmtpSecure" value="false"/>

    <!-- pripadne jmeno uzivatele pro SMTP server -->
    <add key="SmtpUserName" value=""/>

    <!-- pripadne heslo pro uzivatele SMTP serveru -->
    <add key="SmtpPwd" value=""/>

</appSettings>
<connectionStrings>
    <!-- kam se ma server prihlasit k databazi -->
    <!-- Data source ... k jamemu native provideru -->
    <!-- server ... k jakemu serveru -->
    <!-- Integrated Security ... s jakym zabezpecenim -->
    <!-- database ... k jake db -->
    <add name="ConnectionString" connectionString="Data source=Microsoft SQL
    Server;server=MoneyMaker;Integrated Security=SSPI;database=MMDB" />
</connectionStrings>
</configuration>

```

Každou svoji akci *MoneyDownload* zapisuje do logů. Příklad takových logů ukazuje text 2.

Pokud nastane při zpracovávání dat chyba (nepodaří se data stáhnout, zpracovat, či uložit do databáze), *MoneyDownload* odešle email obsahující logy na adresu zadanou v konfiguračním souboru.

Text 2: zkrácený výpis logů MoneyDownload

```
=== Start === 2008-09-30 01_00_01
Downloading ...
http://www.cnb.cz/cs/financni_trhy/devizovy_trh/kurzy_devizoveho_trhu/rok.txt?rok=1991
Parsing and saving...
Memory usage: 1 206 568
=== End step === 2008-09-30 01_03_28
Downloading ...
http://www.cnb.cz/cs/financni_trhy/devizovy_trh/kurzy_devizoveho_trhu/rok.txt?rok=1992
Parsing and saving...
Memory usage: 1 209 040
=== End step === 2008-09-30 01_06_44
...
Downloading ...
http://www.cnb.cz/cs/financni_trhy/devizovy_trh/kurzy_devizoveho_trhu/rok.txt?rok=2008
Parsing and saving...
Memory usage: 1 234 072
=== End step === 2008-09-30 02_26_32
=== End === 2008-09-30 02_26_32
```

Formát logů je následující:

Text 3: formát logů MoneyDownload

```
1 === Start === <datum zahájení celého procesu aktualizace> <čas zahájení celého...>
2 Downloading ...
3 <adresa stránky, která se stahuje>
4 Parsing and saving...
5 Memory usage: <množství spotřebované paměti v MB>
6 === End step === <datum ukončení stahování a parsování dat ze zadané adresy> <čas>
7 ...
8 === End === <datum ukončení celého procesu> <čas>
```

Řádek 1 a 8 jsou v souboru právě jednou – na začátku a na konci. Řádek 7 značí další možné opakování bloku, který se skládá z řádků 2 až 6.

Jak program pracuje

Program pracuje přímočaře – připojí se na web a k databázi, stahuje data a zpracovává je. Podívejme se na příklad stránky, kterou program *MoneyDownload* zpracovává (text 4 níže). Data *MoneyDownload* získal zkopírováním hodnoty `DownloadAddress` z konfiguračního souboru do internetového prohlížeče a nahrazením {0} rokem 2000.

Text 4: Příklad dat, která MoneyDownload zpracovává (čísla 1 – 3 na začátku řádků značí čísla řádků, která v souboru nejsou, slouží nám pouze zde, pro orientaci, kde řádky skutečně končí)

```
1 Datum|1 AUD|1 CAD|1 CHF|1 DKK|1 EUR|1 GBP|100 GRD|100 HUF|100 JPY|1 NOK|1 NZD|1 PLN|1
SEK|100 SIT|100 SKK|1 USD|1 XDR|1 ATS|100 BEF|1 DEM|100 ESP|1 FIM|1 FRF|1 IEP|1000 ITL|100
LUF|1 NLG|100 PTE
```

```

2 03.01.2000|23,363|24,622|22,458|4,849|36,100|57,807|10,937|14,183|34,950|4,474|18,631|
8,632|4,216|18,152|85,164|35,822|49,111|2,623|89,490|18,458|21,697|6,072|5,503|45,838|
18,644|89,490|16,381|18,007
3 04.01.2000|23,409|24,638|22,451|4,842|36,035|57,601|10,929|14,152|34,957|4,458|18,584|
8,648|4,206|18,113|85,161|35,602|48,809|2,619|89,328|18,424|21,657|6,061|5,494|45,755|
18,611|89,328|16,352|17,974
...

```

V prvním řádku dat je hlavička. Ta popisuje formát následujících dat. V každém řádku jsou pole oddělená znakem |. První pole obsahuje datum, další hodnotu, kolik českých korun se musí zaplatit, abychom dostali dané množství dané cizí měny. O kterou měnu se jedná, je patrné z hlavičky. Co řádek, to nový den.

Hlavička je zpracovávána funkcí `parseHeadLine`, další (datové) řádky funkcí `ParseDataLine` (zpracuje a uloží do databáze). Logovací řetězec a email zasílaný při chybě sestavuje funkce `info`.

☞ Související témata

- ☒ Tabulky v databázi pro ukládání dat o měnách (strana 69).

SportsDownload

Pravidelně stahuje z webu www.liga.cz na projektový server data týkající se plánovaných zápasů a výsledků již uskutečněných zápasů.

Spouští ho *Task Scheduler*. Zdrojový kód je umístěn v `cribber\SportsDownload`.

Konfigurační soubor pro tento program musí být umístěn ve stejném adresáři jako samotný spustitelný soubor. Ukázkový konfigurační soubor je umístěn v textu 5 (stránka 15). *SportsDownload* má jeden volitelný vstupní parametr `-NOCONSOLE`. Když je tento parametr zadán, *SportsDownload* nevypisuje nic na obrazovku.

Text 5: konfigurační soubor pro SportsDownload

```

<!-- Konfiguracni soubor Cribberu pro MoneyMaker -->
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <appSettings>
    <!-- Adresar pro ukladani logu -->
    <add key="LogDir" value="logs" />
    <!-- Format jmena jednotlivych souboru s logy -->
    <add key="LogFile" value="log {0}.txt" />
    <!-- Fromat data a casu -->
    <add key="DateTimeFormat" value="yyyy-MM-dd hh_mm_ss" />

    <!-- Adresa stranky, na které lze nalezt vysledky pro zadany den -->
    <add key="DownloadAddress" value="http://www.liga.cz/vysledky/?
year={0};month={1};day={2}" />
    <!-- Adresa hlavniho webu -->
    <add key="RootAddress" value="http://www.liga.cz" />
    <!-- Kolik poslednich dni ma byt stazeno -->
    <add key="HistoryDays" value="30" />
    <!-- Kolik nasledujicich dnu nas zajima (rozpisy zapasu) -->

```

```

<add key="FutureDays" value="3" />

<!-- Kolik procent hodnotou nejmensich i nejvetsich sazkovych kurzu zahodim.
Je to opatreni proti nedokonalostem v parsovani - obcas je kurz zadan
v jinem formatu, nez je bezne a je pak precten chybne. V těchto případech byva
vetsinou velmi vysoky, nebo velmi nizky. Promenna RatesThrowOut specifikuje, kolik
procent techto extremich hodnot bude preventivne zahozeno z kazdeho konce extremu. -->
<add key="RatesThrowOut" value="0.1" />

<!-- Udaje pro komunikaci pomoci emailu -->
<add key="MailFrom" value="soft.projekt@seznam.cz"/>
<add key="MailTo" value="petsvec@tiscali.cz"/>
<add key="MailSubject" value="MoneyMaker - Cribber - sports - ERROR"/>
<add key="SmtpServer" value="localhost"/>
<add key="SmtpSecure" value="false"/>
<add key="SmtpUserName" value="-"/>
<add key="SmtpPwd" value="-"/>
</appSettings>

<!-- Udaje pro pripojeni k databazi -->
<connectionStrings>
  <add name="ConnectionString" connectionString="Data source=Microsoft SQL
Server;server=AMD;Integrated Security=SSPI;database=MMDB" />
</connectionStrings>
</configuration>

```

Každou svojí akci *SportsDownload* zapisuje do logů. Příklad takových logů ukazuje text 6.

Pokud nastane při zpracovávání dat chyba (nepodaří se data stáhnout, zpracovat, či uložit do databáze), *SportsDownload* odešle email obsahující logy na adresu zadanou v konfiguračním souboru.

Text 6: zkrácený výpis logů *SportsDownload*

```

=== Start === 2009-02-05 01_00_01
Downloading ... http://www.liga.cz/vysledky/?year=2009&month=1&day=1
Parsing and saving...
/hokej/usa/ahl-2008-2009/matchdetails.php?matchid=6425030
/fotbal/anglie/anglie-conference-2008-2009/matchdetails.php?matchid=6389619
/fotbal/anglie/anglie-conference-2008-2009/matchdetails.php?matchid=6389620
/fotbal/anglie/anglie-conference-2008-2009/matchdetails.php?matchid=6389622
...
/fotbal/severni-irsko/1-severoirska-liga-2008-2009/matchdetails.php?matchid=6379539
Memory usage: 1 113 656
=== End step === 2009-02-05 01_00_15
Downloading ... http://www.liga.cz/vysledky/?year=2009&month=1&day=2
Parsing and saving...
/hokej/usa/ahl-2008-2009/matchdetails.php?matchid=6425031
/hokej/usa/ahl-2008-2009/matchdetails.php?matchid=6425032
...
=== End === 2009-02-05 01_03_19

```

Formát logů je následující:

Text 7: formát logů SportsDownload

```
1 === Start === <datum zahájení celého procesu aktualizace> <čas zahájení celého...>
2 Downloading ... <adresa stránky, která se stahuje>
4 Parsing and saving...
5 <relativní adresa stránky obsahující podrobná data k určitému zápasu>
6 <relativní adresa stránky obsahující podrobná data k určitému zápasu>
...
7 Memory usage: <množství spotřebované paměti v MB>
8 === End step === <datum ukončení stahování a parsování dat pro daný den> <čas>
...
9 === End === <datum ukončení celého procesu> <čas>
```

Řádek 1 a 9 jsou v souboru právě jednou – na začátku a na konci. Mezi řádky 6 a 7 pokračuje seznam dalších stahovaných stránek. Každá se týká jednoho zápasu. Mezi řádky 8 a 9 se opakují sekvence pro další dny (podobné řádkům 2 až 8).

Jak program pracuje

Seznámení s prostředím

I *SportsDownload*, *cribber* pro stahování sportů pracuje přímočaře – připojí se na web a k databázi, stahuje data a zpracovává je. Celá procedura je složitější, než u zpracování dat o měnách – to je dáno charakterem a množstvím dat.

The screenshot shows the LIGA.CZ website interface. At the top, there are links to various sports and betting services. The main navigation bar includes links for HOME, FOTBAL, HOKEJ, BASKETBAL, and HÁZENÁ. Below this, there is a sidebar with a list of links such as DNEŠNÍ ZÁPASY, VÝSLEDKY, DISKUZE, TIPY, TIPY UŽIV. (3), LIVESCORE, ZPRÁVY, SOUTĚŽ, OBLÍBENÉ, MŮJ TIKET, SÁZENÍ ONLINE, ODKAZY, NASTAVENÍ, SOFTWARE, and KONTAKT. The main content area displays 'Včerejší výsledky' (Yesterday's results) for the date 04.02.2009. It features a table of sports results for AHL and Albánská liga. To the right of the table is a calendar for February 2009, highlighting the 4th. Below the table, there is a search bar for finding a team.

Všechny sporty			
Fotbal			
04.02.2009			
AHL			
23:55	Hartford Wolf Pack - Manchester Monarchs	3:6	(1:1, 1:3, 1:2)
	Portland Pirates - Worcester Sharks	3:5	(3:0, 0:2, 0:3)
	Binghamton Senators - San Antonio Rampage	4:3	(1:0, 3:2, 0:1)
	Rochester Americans - Rockford IceHogs	1:2	(0:1, 0:0, 1:1)
	Hamilton Bulldogs - Syracuse Crunch	1:2	(0:0, 1:0, 0:2)
	Chicago Wolves - Peoria Rivermen	5:1	(1:0, 2:1, 2:0)
	Quad City Flames - Grand Rapids Griffins	2:3	(0:2, 2:0, 0:1)
Albánská liga			
15:00	Apolonia Fier - Bylis Ballsh	1:0	(0:0, 1:0)
	Lushnja KS - Elbasani SK	1:0	(1:0, 0:0)
	Vllaznia Shkodër - Besa Kavajë	4:1	(0:1, 4:0)
	Partizani Tirana - Dinamo Tirana KS	1:0	(0:0, 1:0)
	Flamurtari Vlorë - Teuta Durrës	1:1	(1:1, 0:0)
05.02.2009			

Obrázek 2: Přehled sportovních výsledků z jednoho dne

Opět se podívejme na příklad stránky, kterou program *SportsDownload* zpracovává (text 8 níže). Příklad je spíše ilustrativní, zahrnuje pouze malou část dat a jednu z několika možností, co se týká typu obsažených dat. Data jsme získali zkopírováním hodnoty proměnné `DownloadAddress`, která má aktuálně hodnotu `http://www.liga.cz/vysledky/?year={0};month={1};day={2}`, z konfiguračního souboru do internetového prohlížeče a dosazením konkrétních údajů pro den 4.1.2009: `http://www.liga.cz/vysledky/?year=2009&month=1&day=4` (Obrázek 2). Nepodstatné části stránky jsou vynechané:

Text 8: Příklad dat, která SportsDownload zpracovává (čísla 1 – 2 na začátku řádků značí čísla řádků, která v datech nejsou, slouží nám pouze zde, pro orientaci, kde řádky skutečně končí)

```
...
1 <tr class="rtitle flags"><th colspan="4" class="left"><a class="cal fl37"
href="/hokej/usa/ahl-2008-2009/">AHL</a></th></tr>
<tr class="first-row"><td class="time first-cell">23:55</td><th><a
href="/hokej/usa/ahl-2008-2009/matchdetails.php?matchid=6425235" onclick="return !
window.open(this.href, '', 'height=500,width=500,scrollbars=yes,statusbar=no,menubar=no,rez
ze=yes');">Hartford Wolf Pack&nbsp;-&nbsp;Manchester Monarchs</a></th><td
class="score"><strong>3:6</strong></td><td class="sresult last-
cell">(1:1,&nbsp;1:3,&nbsp;1:2)</td></tr>
2 <tr class=" strong"><td class="time first-cell">&nbsp;</td><th><a
href="/hokej/usa/ahl-2008-2009/matchdetails.php?matchid=6425236" onclick="return !
window.open(this.href, '', 'height=500,width=500,scrollbars=yes,statusbar=no,menubar=no,rez
ze=yes');">Portland Pirates&nbsp;-&nbsp;Worcester Sharks</a></th><td
class="score"><strong>3:5</strong></td><td class="sresult last-
cell">(3:0,&nbsp;0:2,&nbsp;0:3)</td></tr>
...
```

Zaměřme se na první řádek (ostatní jsou jeho obdobou). Z adresy odkazu (`href="/hokej/usa/ahl-2008-2009/"`) získáváme druh sportu (hokej) a název soutěže (AHL). Dále vidíme čas zápasu (23:55), odkaz na stránku s detaily (`/hokej/usa/ahl-2008-2009/matchdetails.php?matchid=6425235`), jména zúčastněných týmů a výsledek. Pro další podrobnosti o zápasu si *SportsDownload* musí stáhnout data z uvedené adresy. Podívejme se na ní: `http://www.liga.cz/hokej/usa/ahl-2008-2009/matchdetails.php?matchid=6425235` (Obrázek 3, Text 9).

Hokej » USA » AHL 2008/2009

04.02.2009

Hartford Wolf Pack - Manchester Monarchs

3 : 6

(1:1, 1:3, 1:2)

Kursy 1X2 (15) Under/Over (4) Asijský handicap (1) Moneyline kursy (1)

Sázkové kanceláře: 15	1	0	2	10	02
Bet-At-Home [□] (www)	1.80	4.30	2.90	0.00	0.00
BetClick [□] (www)	1.75	4.25	3.10	0.00	0.00
Betoto [□] (www)	1.70	4.45	3.30	0.00	0.00
Bwin.com [□] (www)	1.80	4.20	3.10	0.00	0.00
Chance [□] (ČR)	1.85	4.20	3.00	1.28	1.74
Doxx [□] (SK)	1.79	3.80	3.44	1.22	1.81
Fortuna [□] (ČR)	1.80	4.00	3.27	1.26	1.80
Gamebookers [□] (www)	1.80	4.20	3.10	1.26	1.78
Nike [□] (SK)	1.80	4.00	2.90	1.28	1.80
PartyBets [□] (www)	1.80	4.20	3.10	0.00	0.00
Terno [□] (SK)	1.80	4.00	3.27	1.26	1.80
Tipsport [□] (ČR)	1.79	3.80	3.44	1.22	1.81
TipTipTip [□] (ČR)	1.81	3.90	3.31	1.26	1.79
Victoria-Tip [□] (ČR)	1.80	3.90	3.34	1.23	1.80
Victory Tip [□] (SK)	1.80	4.25	3.10	1.29	1.84

[zavřít okno](#)

Obrázek 3: Detail zápasu

Obrázek 3: Detail zápasu

Text 9: Příklad dat, která SportsDownload zpracovává

```
...
<tr class=" strong"><th class="first-cell"><a
href="/bookmaker/258/http://www.betbrain.com/common/dlog.jsp?
dealerId=3000336&site=224" onclick="window.open(this.href);return false"
class="elink">BetClick</a> <span>(www)</span></th><td class="course
inc"><span>1.75</span></td><td class="course inc"><span>4.25</span></td><td class="course
inc"><span>3.10</span></td><td class="split"></td><td class="course
inc"><span>0.00</span></td><td class="course inc last-cell"><span>0.00</span></td></tr>
<tr class=""><th class="first-cell"><a
href="/bookmaker/194/http://www.betbrain.com/common/dlog.jsp?
dealerId=3000255&site=224" onclick="window.open(this.href);return false"
class="elink">Betoto</a> <span>(www)</span></th><td class="course inc"><span>1.70</span></td>
<td class="course best-betrate inc"><span>4.45</span></td><td class="course
inc"><span>3.30</span></td><td class="split"></td><td class="course
inc"><span>0.00</span></td><td class="course inc last-cell"><span>0.00</span></td></tr>
...
```

Z této stránky získáváme data vypsaných kurzů jednotlivými sázkovými kancelářemi.

Algoritmus

Program se připojí k databázi, připraví si soubor pro ukládání logů a klienty pro připojení k webu. Potom pro každé datum ze zadaného rozmezí stáhne data a zpracuje je – stránku si převede do XML formátu a dále zpracovává metodou SAX.

V řádku týkajícím se daného zápasu rozeznává zadaná slova (sporty): HOKEJ, FOTBAL, BASKETBAL a HAZENA. Dále vyhledá řetězec obsahující název soutěže, názvy obou zúčastněných týmů, ročník soutěže, datum zápasu a zda se jedná o soutěž mužů, nebo žen. Pokud zjistí, že daný zápas

ještě není v databázi uložen, přidá ho. Pokud je zápas již odehraný, aktualizuje v databázi výsledky.

Poté zkoumá detaily zápasu. Stáhne stránku s detaily a vybere z ní vypsané kurzy. Ty zprůměruje a uloží do databáze.

Na závěr akce odešle *SportsDownload* informační email, pokud nastala chyba.

Poznámky

Na první pohled se může zdát, že v tabulkách zápasů databáze (T405Matches, strana 90) je nepřiměřeně velké množství hodnot výsledků zápasu *NULL* a že *SportsDownload* nepracuje správně. Data jsme pečlivě zkoumali a zjistili jsme, že zápasy s nevyplněným výsledkem byly přesunuty na jiné dny, byl změněn jejich čas konání a podobně. Důvody jsou různé, např. tým onemocněl, počasí se výrazně zhoršilo oproti předpokladům atd. Také se stává, že tým odkoupí jiný investor a jméno týmu se změní. Proto je v databázi relativně vysoké procento zápasů bez výsledku – měly být odehrány, ale nakonec nebyly.

Občas se dokonce stává, že data jsou chybně uvedena i na webu www.liga.cz, odkud jsou stahovány – v takovém případě (pokud se nám podaří ho rozpoznat), jsou data opět zahozena a nejsou ukládána do databáze.

Související témata

- ☒ Tabulky v databázi pro ukládání dat o sportovních výsledcích (strana 84).

BusinessDemo a Businessman

Businessman má své zdrojové kódy uloženy ve stejnojmenném adresáři. Jedná se o klíčovou část projektu, která obstarává každodenní spouštění pluginů a ukládání jejich odhadů do databáze. *Businessman* potřebuje ke svému běhu databázi a denně aktualizovaná data kurzů měn, plánovaných sportovních utkání a jejich výsledků.

BusinessDemo má dvě verze – jednu pro testování sportovních pluginů a druhou pro testování pluginů pro odhadování kurzů měn. Je uloženo v adresářích *businessdemoMoney* a *businessdemosport*. Každá část se skládá ze dvou hlavních pod-částí. První část je obecná, obsahující základní třídy, stejné, jako používá i *Businessman* běžící na serveru. Tato část se nachází v podadresáři *businessdemo*\BusinessmanCommon*. Druhá část pouze využívá tyto třídy k tomu, aby rychle a jednoduše získala výsledky od uživatelem vytvořeného pluginu na ukázkových datech. Data nejsou přímou součástí programu *BusinessDemo*, ale pro běh programu jsou nutná.

Businessdemo spouští vždy právě jednu instanci strategie, zatímco *Businessman* spouští postupně za sebou více instancí každé strategie (a strategie samotné běží paralelně). *Businessman* pracuje s databází, zatímco *Businessdemo* pouze s textovými soubory (vstupní a výstupní textový soubor). *BusinessmanCommon* jsou objekty sloužící instanci k přístupu k datům a zároveň ji umožňují komunikovat s vnějším prostředím – zapisovat do logů, ukládat výsledky atd.

Businessman

Základní třídy pro *BusinessDemo* stejně jako pro *Businessman* jsou *MoneyPack* a *SportsPack*.

MoneyPack

Tuto třídu využívají *Businessman* a *BusinessdemoMoney*.

Základní schéma třídy ilustruje obrázek 4. Pro přehlednost byly některé funkce a hodnoty, které nejsou až tak podstatné, vynechány.

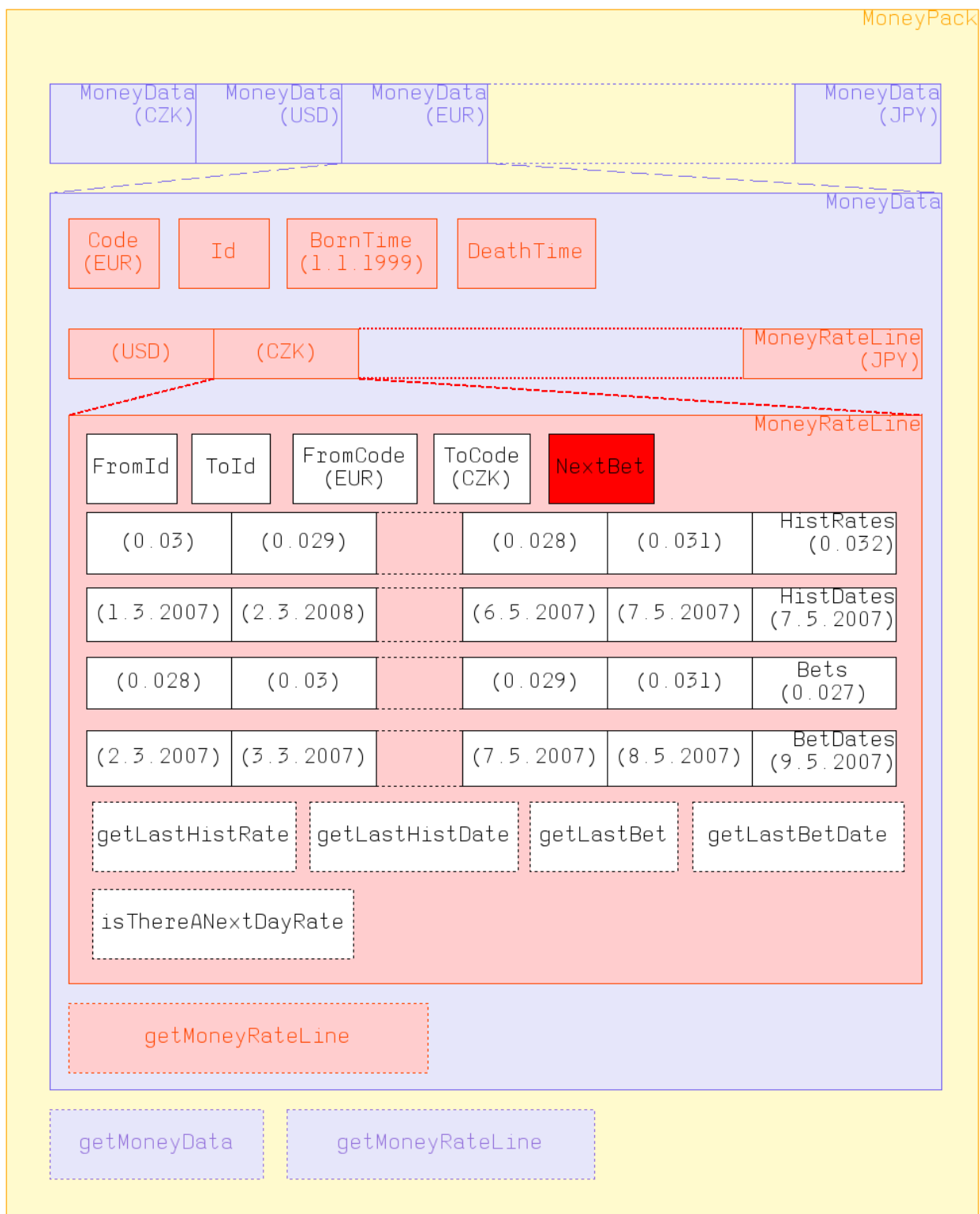
Třída *MoneyPack* je definována v souboru `businessman\BusinessmanCommon\Money\MoneyPack.cs`. Třída *MoneyPack* dědí ze třídy *PackBase* definované v souboru `businessman\BusinessmanCommon\businessman\BusinessmanCommon\Money\PackBase.cs` pro *Businessman* (ta zajišťuje komunikaci s SQL databází).

Do třídy *MoneyPack* patří mimo jiné:

Položka	Význam
slovník <i>Money</i>	Slovník obsahující třídu <i>MoneyData</i> (viz. níže) pro každou uvedenou měnu, tedy klíčem slovníku je měna a hodnotou třída <i>MoneyData</i> .
interní funkce <i>saveAllBets</i>	Hlavním účelem funkce je vytisknout (do souboru nebo databáze) všechny tipy spočítané pluginem a skutečné hodnoty, které následovaly, pro všechny dny. Funkce pracuje se všemi daty z <i>Money</i> , tedy tiskne všechny tipy a kurzy mezi všemi měnami, které byly tipovány (kurzy, které plugin netipoval, vynechává).
interní funkce <i>reset</i>	Zavolá funkce <i>reset</i> pro všechny položky slovníku <i>Money</i> . V zásadě se jedná o vynulování všech dat slovníku.
funkce <i>getMoneyData</i>	Tato funkce slouží k nalezení struktury <i>MoneyData</i> ve slovníku pomocí řetězce označujícího hledanou měnu, např. „USD“, „EUR“ atd. Funkce vrací instanci třídy odpovídající dané měně.
funkce <i>getMoneyRateLine</i>	Funkce vrací data popisující kurzy mezi dvěma zadanými měnami.

Než podrobněji probereme slovník *Money*, prosím, podívejte se na již zmíněný obrázek 4 na straně 22 – pak by mělo být jasněji zřetelné, jak celý *MoneyPack* funguje. Na obrázku v závorkách jsou možné ukázkové hodnoty pro ještě ilustrativnější znázornění třídy *MoneyData*.

Ve slovníku *Money* jsou uložena data pro každou měnu – každá měna zde má svůj *MoneyData* záznam. Popišme si nyní třídu *MoneyData* podrobněji. Kód třídy je uložen v souboru `businessman\BusinessmanCommon\Money\MoneyData.cs`.



Obrázek 4: Hrubé schéma třídy MoneyPack

Každá instance třídy `MoneyData` má mimo jiné:

Položka	Význam
<code>Id</code>	Identifikační číslo.
<code>Code</code>	Kódem se rozumí řetězec obsahující název měny, např. „JPY“, „CZK“...
<code>BornTime</code>	Datum, kdy měna vznikla – fakticky je to datum, od kterého se vyskytuje v tabulkách kurzů vydávaných ČNB.
<code>DeathTime</code>	Datum, kdy měna zanikla, pokud již zanikla (pokud ještě nezanikla, je vyplněno implicitní hodnotou nejnižšího možného data).
slovník <code>RateLines</code>	Slovník obsahující třídu <code>MoneyRateLine</code> (viz. níže) pro každou uvedenou měnu, tedy klíčem slovníku je měna a hodnotou třída <code>MoneyRateLine</code> znázorňující historii vzájemného kurzu obou měn.
funkce <code>getMoneyRateLine</code>	Tato funkce slouží k nalezení struktury <code>RateLine</code> ve slovníku pomocí řetězce označujícího hledanou měnu, např. „USD“, „EUR“ atd. Funkce vrací instanci třídy odpovídající dané měně.

MoneyRateLine

`MoneyRateLine` je poslední vnořená třída ve třídě `MoneyPack`. Ve slovníku `MoneyRateLine` jsou uložena data pro každou měnu – každá měna zde má svůj `MoneyRateLine` záznam obsahující záznamy kurzů mezi ní a dalšími měnami. Kód třídy je uložen v souboru `businessman\BusinessmanCommon\Money\MoneyRateLine.cs`.

Každá instance třídy `MoneyRateLine` obsahuje mimo jiné:

Položka	Význam
<code>FromId, ToId</code>	Id instancí „ze kterých“ je kurz uložen „na které“.
<code>FromCode, ToCode</code>	Kódem se rozumí řetězec obsahující název měny, např. „JPY“, „CZK“... <code>FromCode</code> a <code>ToCode</code> mají stejný význam jako <code>FromId</code> a <code>ToId</code> s tím rozdílem, že informace je uložena ve formě řetězce („CZK“) místo číselného Id.
<code>list HistRates</code>	Seznam číselných záznamů o kurzech mezi měnou <code>FromCode</code> a měnou <code>ToCode</code> .
<code>list HistDates</code>	Seznam spárovatelný se seznamem <code>HistRates</code> obsahující data (míněno jako datum), kdy kurzy uložené v <code>HistRates</code> nastaly.
<code>list Bets</code>	Seznam obsahující tipy spočítané pluginem.
<code>list BetsDates</code>	Seznam spárovatelný se seznamem <code>Bets</code> obsahující data (míněno jako datum), kdy byly uloženy tipy uložené v <code>Bets</code> .
funkce <code>getLastBet</code>	Funkce vracějící hodnotu posledního uloženého tipu.
funkce <code>getLastBetDate</code>	Funkce vracějící datum posledního uloženého tipu.
<code>NextBet</code>	Jediná pro plugin zapisovatelná hodnota celé třídy <code>MoneyPack</code> . Do ní plugin ukládá svůj vypočtený tip. Když je tip zapsán, automaticky se zkopíruje do ostatních struktur (<code>Bets</code> , <code>BetsDates</code>).
funkce <code>isThereANextDayRate</code>	Funkce odpovídající na otázku, zda má plugin počítat i nový tip. Některé dny totiž nemá smysl nový tip počítat, například v neděli, kdy se kurzy nemění.

SportPack

Tuto třídu využívá *Businessman* a *BusinessdemoSport*.

Terminologie

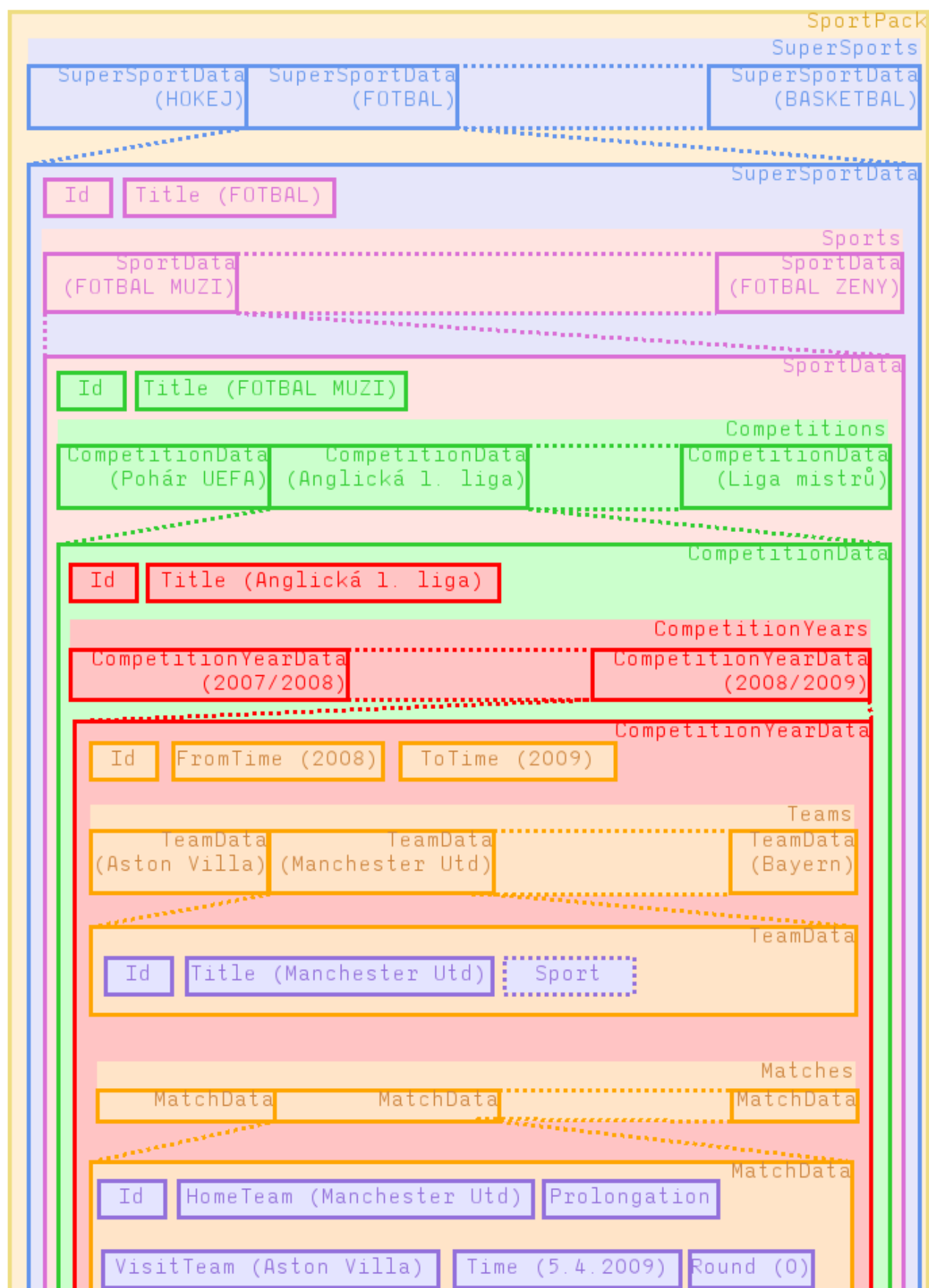
Než si představíme hlavní datovou strukturu, je třeba se seznámit s terminologií, protože v některých případech není tak intuitivní, jako tomu bylo u třídy MoneyPack.

<i>Super sport</i>	Tímto termínem označujeme hlavní jméno sportu, tedy např. fotbal, hokej, basketbal.
<i>Sport</i>	<i>Sport</i> je podmnožinou <i>super sportu</i> , je jím např. ženský fotbal, mužský fotbal, ženský hokej, mužský hokej, mužský hokej atd.
<i>Soutěž, competition</i>	Pod <i>soutěží</i> rozumíme skutečně soutěž, tedy např. Pohár UEFA, 1. anglická liga, NHL, NBA... Soutěž je podmnožinou <i>sportu</i> .
<i>Ročník soutěže, competition year</i>	Jedná se o ročník značený jedním, nebo dvěma letopočty. Např. 2008/2009, nebo jenom 2009. <i>Ročník soutěže</i> je podmnožinou <i>soutěže</i> .
<i>Tým, team</i>	Každý <i>tým</i> má své jméno a sport, kterému se věnuje. Např. Sparta. Sparta je ale v našem systému dvakrát – první je spojena s hokejovou soutěží, druhá s fotbalovou.
<i>Zápas, match</i>	Tento termín lze opět chápat zcela intuitivně, jedná se o zápas dvou týmů, který má (bude mít) nějaký výsledek, má datum konání, týmy, které se ho účastní proti sobě, atd.

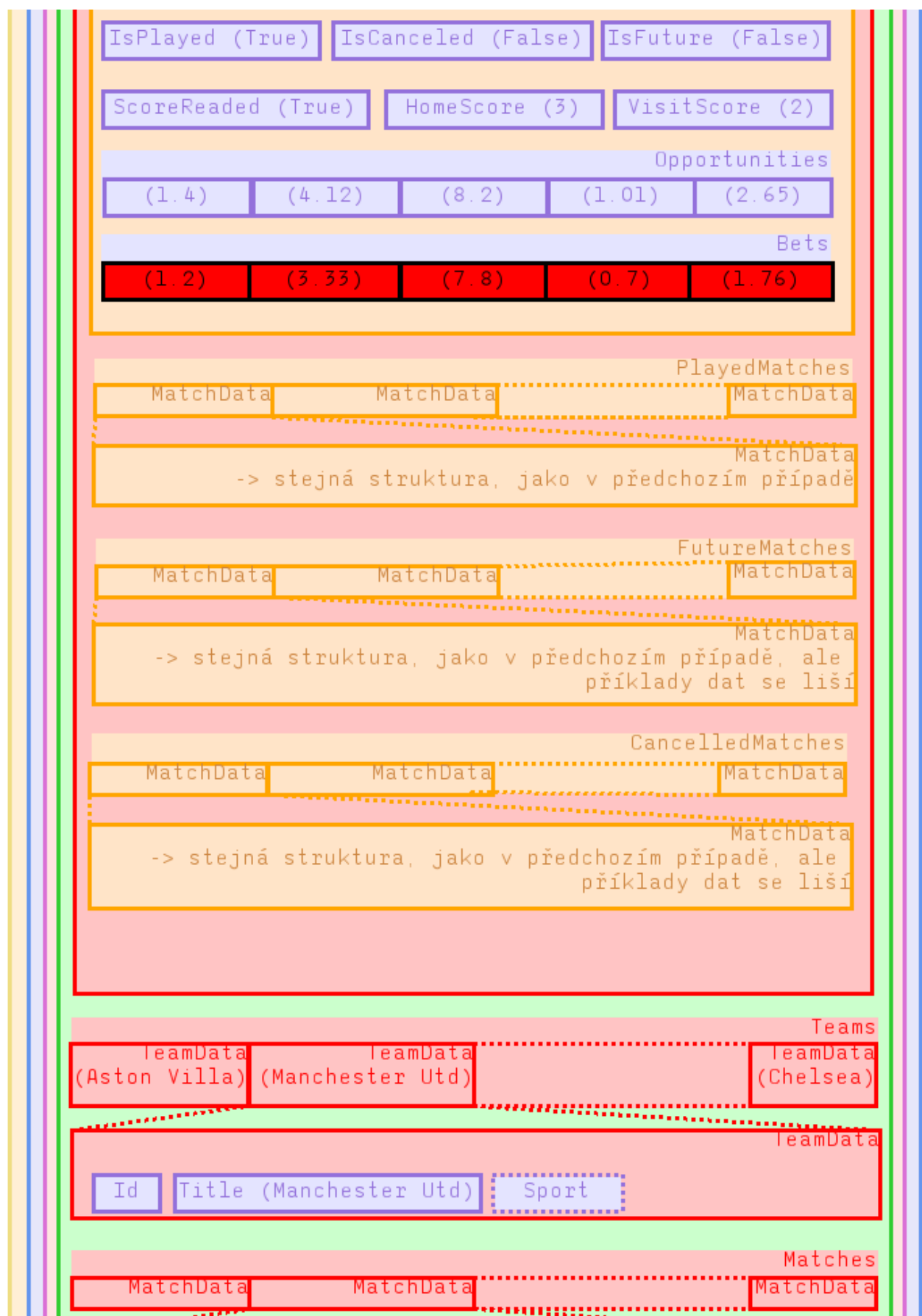
Struktura

Základní schéma ilustrují obrázky 5, 6 a 7 na stranách 25 až 27. Pro přehlednost byly některé funkce a hodnoty, které nejsou až tak podstatné, vynechány. Obrázek se může zdát složitý, nicméně tak složitě to není. Struktura v podstatě kopíruje strukturu databáze, proto je vhodné si prostudovat také databázové tabulky týkající se sportovního sázení, aby bylo vše dobře pochopeno. Obrázky jsou popsány v textu, který za nimi následuje.

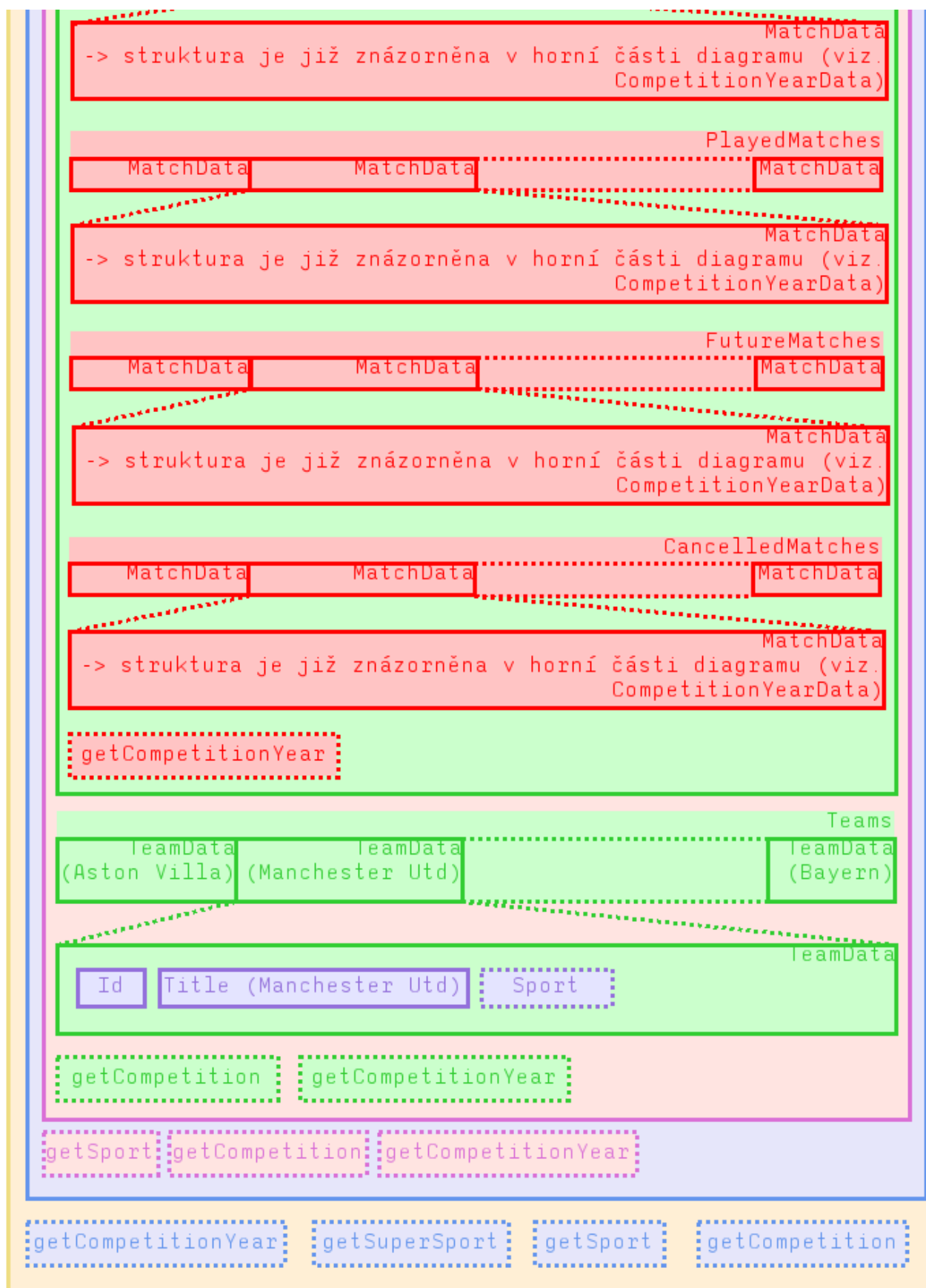
Třída SportPack je definována v souboru businessman\BusinessmanCommon\Sport\SportPack.cs. Třída SportPack dědí ze třídy PackBase definované v souboru businessman\BusinessmanCommon\PackBase.cs (ta zajišťuje komunikaci se SQL databází).



Obrázek 5: Základní struktura SportPack, část 1



Obrázek 6: Základní schéma SportPack, část 2



Obrázek 7: Základní schéma SportPack, část 3

Třída SportPack

Funkčnost a struktura třídy SportPack je obdobná, jako u třídy MoneyPack. Zde je stručný popis nejdůležitějších proměnných a funkcí.

Položka	Význam
slovník SuperSports	Slovník obsahující třídu SuperSportData (viz. níže) pro každý uvedený super sport. Tedy klíčem slovníku je super sport (HOKEJ) a hodnotou třída SuperSportData.
interní funkce saveAllBets	Hlavním účelem funkce je vytisknout (do souboru nebo databáze) všechny tipy spočítané pluginem a skutečné hodnoty, které následovaly, pro všechny dny. Funkce pracuje se všemi tipy, které plugin spočítal. Kurzy pro všechny zápasy, které byly tipovány (kurzy, které plugin netipoval, vynechává).
interní funkce reset	Zavolá funkce reset pro všechny položky slovníku SuperSports. V zásadě se jedná o vynulování všech dat slovníku.
funkce getSuperSport	Tato funkce slouží k nalezení struktury SuperSportData ve slovníku pomocí řetězce označujícího hledaný super sport, např. „HOKEJ“, „FOTBAL“ atd. Funkce vrací instanci třídy odpovídající danému super sportu.
funkce getSport	Tato funkce slouží k nalezení struktury SportData ve slovníku pomocí řetězce označujícího hledaný sport, např. „HOKEJ ZENY“, „FOTBAL MUZI“ atd. Funkce vrací instanci třídy odpovídající danému sportu.
funkce getCompetition	Tato funkce slouží k nalezení struktury CompetitionData ve slovníku pomocí řetězce označujícího hledanou soutěž, např. „NHL“, „Liga mistrů“ atd. Funkce vrací instanci třídy odpovídající dané soutěži.
funkce getCompetitionYear	Tato funkce slouží k nalezení struktury CompetitionYearData ve slovníku pomocí řetězce označujícího hledanou soutěž a její ročník, např. „Liga mistrů 2008/2009“ atd. Funkce vrací instanci třídy odpovídající danému ročníku soutěže.

Třída SuperSportData

Položka	Význam
proměnná Id	Každá instance této třídy má své identifikační číslo Id.
proměnná Title	Každá instance této třídy má své jméno, např. FOTBAL.
slovník Sports	Slovník obsahující třídu SportData (viz. níže) pro každý uvedený sport. Tedy klíčem slovníku je sport (HOKEJ MUZI) a hodnotou třída SportData.

Další funkce mají stejný význam, jako v předchozí třídě.

Třída SportData

Položka	Význam
proměnná Id	Každá instance této třídy má své identifikační číslo Id.
proměnná Title	Každá instance této třídy má své jméno, např. FOTBAL ZENY.
slovník Competitions	Slovník obsahující třídu CompetitionData (viz. níže) pro každou uvedenou soutěž. Tedy klíčem slovníku je <i>soutěž</i> (NHL) a hodnotou třída CompetitionData.
slovník Teams	Slovník obsahující třídu TeamData (viz. níže) pro každý zúčastněný tým. Tedy klíčem slovníku je jméno <i>týmu</i> (Chelsea) a hodnotou třída TeamData.

Zbylé funkce mají opět stejnou funkčnost, jako u třídy SportPack.

Třída CompetitionData

Třída pro práci s daty o jednotlivých soutěžích.

Položka	Význam
proměnná Id	Každá instance této třídy má své identifikační číslo Id.
proměnná Title	Každá instance této třídy má své jméno, např. „1. Anglická liga“.
slovník CompetitionYears	Slovník obsahující třídu CompetitionYearData (viz. níže) pro každý ročník soutěže. Tedy klíčem slovníku je <i>ročník soutěže</i> (NHL 2008/2009) a hodnotou instance třídy CompetitionYearData.
pole Teams	Pole obsahující instance třídy TeamData (viz. níže) pro každý zúčastněný tým.
pole Matches	Pole obsahující instance třídy MatchData (viz. níže) pro každý zápas spadající do ročníku soutěže.
pole PlayedMatches	Pole obsahující instance třídy MatchData (viz. níže) pro každý zápas odehraný v rámci ročníku soutěže.
pole FutureMatches	Pole obsahující instance třídy MatchData (viz. níže) pro každý zápas plánovaný v rámci ročníku soutěže.
pole CanceledMatches	Pole obsahující instance třídy MatchData (viz. níže) pro každý zápas v rámci ročníku soutěže, který byl zrušen.

Zbylé funkce mají opět stejnou funkčnost, jako u třídy SportPack.

Třída CompetitionYearData

Položka	Význam
proměnná Id	Každá instance této třídy má své identifikační číslo Id.
proměnná FromTime	Rok, kdy ročník soutěže začal.
proměnná ToTime	Rok, kdy ročník soutěže skončil (může se shodovat s FromTime)
pole Teams	Pole obsahující instance třídy TeamData (viz. níže) pro každý zúčastněný tým.
pole Matches	Pole obsahující instance třídy MatchData (viz. níže) pro každý zápas spadající do ročníku soutěže.
pole PlayedMatches	Pole obsahující instance třídy MatchData (viz. níže) pro každý zápas odehraný v rámci ročníku soutěže.
pole FutureMatches	Pole obsahující instance třídy MatchData (viz. níže) pro každý zápas plánovaný v rámci ročníku soutěže.
pole CanceledMatches	Pole obsahující instance třídy MatchData (viz. níže) pro každý zápas v rámci ročníku soutěže, který byl zrušen.

Zbýlé funkce mají opět stejnou funkčnost, jako u třídy SportPack.

Třída TeamData

Položka	Význam
proměnná Id	Každá instance této třídy má své identifikační číslo Id.
proměnná Title	Jméno týmu (např. Bohemians Praha).
Funkce sport	Funkce vracející <i>sport</i> , ve kterém se tým angažuje.

Zbýlé funkce mají opět stejnou funkčnost, jako u třídy SportPack.

Třída matchData

Položka	Význam
proměnná id	Každá instance této třídy má své identifikační číslo id.
proměnná homeTeam	Tým domácích
proměnná visitTeam	Tým hostů
proměnná prolongation	Příznak, zda byl zápas prodlužován
proměnná Time	Datum a čas konání zápasu
proměnná round	Číslo kola v rámci soutěže
proměnná isPlayed	Příznak, zda je již zápas odehraný
proměnná isCanceled	Příznak, zda byl zápas zrušen
proměnná isFuture	Příznak, zda se zápas teprve bude konat
proměnná scoreReaded	Příznak, zda je již načtené skóre do systému
proměnná homeScore	Skóre domácích
proměnná visitScore	Skóre hostů
pole opportunities	Pole má pevnou délku 5 a slouží pro uložení kurzů na zápas, jak je stanovily sázkové kanceláře
pole bets	Pole má pevnou délku 5 a slouží pro uložení tipů na kurzy na zápas, jak plugin očekává, že je stanoví sázkové kanceláře

Zbylé funkce mají opět stejnou funkčnost, jako u třídy SportPack.

Core

Soubor: businessdemo*\BusinessmanCommon\Core.cs

Třída obsahující základní funkčnost, jako třeba logovací funkce.

PluginBase

Soubor: businessdemo*\BusinessmanCommon\PluginBase.cs

Abstraktní třída obsahující připojení k databázi, připojení k pluginu (.Net Assembly), odkaz na rodiče třídy atd. Definuje, jak budou objekty pluginů vypadat.

BusinessDemo

BusinessDemo tvoří dva soubory – jeden pro testování pluginů zabývajících se odhadem kurzů měn a druhý pro testování sportovních pluginů.

Hlavní soubory obsahující kód jsou uloženy v businessdemo*\BusinessDemo\Program.cs.

Co je úkolem BusinessDemo

Úkolem *BusinessDemo* je chovat se jako *Businessman*, aby si mohl uživatel ověřit, že jeho plugin neobsahuje chyby. Zároveň je tak jednoduchý, že si ho uživatel může jednoduše spustit doma bez složité instalace a konfigurace. *BusinessDemo* potřebuje ke svému běhu uživatelův

plugin a soubor s daty (o kurzech nebo sportech). Na rozdíl od *Businessman* nespouští plugin každý den jednou, ale spustí ho na všechny dny zadané v datovém souboru postupně hned za sebou. Při tom se samozřejmě pro plugin tváří stejně jako *Businessman*. Na rozdíl od aplikace *Businessman* si samo nestahuje aktuální data vývoje kurzů nebo sportovních výsledků a pracuje pouze s těmi, které mu již hotové předložíte. Rovněž netvoří více instancí strategie a tedy nekříží proměnné.

Princip práce BusinessDemo

BusinessdemoMoney

Nejprve načte plugin k otestování a připraví si soubor pro zápis výstupu. Vytvoří prázdnou instanci třídy *MoneyPack*, do slovníku *Money* přidá „CZK“. Poté vytvoří instanci pluginu.

Následuje načítání datového souboru. Nejprve načte hlavičku, aby věděl, v jakém pořadí data následují. Poté načte první den a získaná data se uloží do objektu třídy *MoneyPack*. Spustí se plugin, očekává se, že provede odhad a uloží ho. Uložená hodnota se zpracuje a do *MoneyPack* se načtou hodnoty z druhého načteného dne. Pak se opět spustí plugin a tak dále, dokud se nenačte poslední den.

Nakonec *BusinessdemoMoney* zavře otevřené soubory (datový a výstupní) a skončí.

BusinessdemoSport

Businessdemo pro sporty pracuje velmi podobným způsobem.

Nejprve načte plugin k otestování a připraví si soubor pro zápis výstupu. Vytvoří prázdnou instanci třídy *SportPack* a instanci pluginu.

Program začíná načítáním datového souboru. Načte prvních pět dní dat, ale označí je jako ještě nehrané, aby plugin věděl, jaké zápasy má tipovat. Tyto hodnoty uloží do objektu třídy *SportPack*. Spustí se plugin, očekává se, že provede odhad a uloží ho. Uložená hodnota se zpracuje a do *SportPack* se načtou hodnoty z dalšího (šestého) načteného dne, data z prvního dne se označí jako již zhrané zápasy. Pak se opět spustí plugin a tak dále, dokud se nenačte poslední den.

Nakonec *BusinessdemoSport* zavře otevřené soubory (datový a výstupní) a skončí.

Formát datových souborů

Sporty

K *BusinessdemoSport* je přiložen datový soubor obsahující informace o již odehraných zápasech za relativně krátké období. Soubor je uložen v cestě *businessdemosport\SportsData.txt*.

Text 10 ukazuje prvních pět řádků z tohoto souboru. První řádek je hlavička, popisující ostatní řádky. Hlavička je při zpracovávání souboru ignorována – slouží jen pro orientaci uživateli. Hodnoty musejí být v souboru vždy v pevně daném pořadí. Hodnoty jsou na řádcích oddělené pomocí svislítky '|'.

První hodnota je datum, kdy se zápas odehrál, následuje jméno sportu, pohlaví hráčů, jméno soutěže, pod kterou zápas spadal, další dvě hodnoty obsahují ročník soutěže, následuje

jméno domácího týmu a týmu hostů, jejich skóre (nejdříve domácí, pak hosté), příznak, zda byl zápas prodlužován, kolikáté kolo v soutěži se zápas hrál a na závěr pětice kurzů vypsanych sázkovými kanceláři (výhra domácích, ne-prohra domácích, remíza, ne-prohra hostů, výhra hostů).

Soubor může takových hlavičkových řádek obsahovat libovolně mnoho s libovolným obsahem (tedy ne jen popisem), řádky jsou ve výsledku ignorovány. Jediným pravidlem je, že takové řádky musí začínat znakem '#'.

Text 10: Ukázka z datového souboru, který obsahuje vývoj kurzů mezi českou korunou a ostatními měnami

```
# Match Date (yyyy/mm/dd)|Sport|Sex|Competition Name|Year From|Year To|Home team Name|
Visitors Team Name|Home Score|Visitor Score|Prolongation|Round|Home will win|Home won't
lose|Draw game|Visitors won't lose|Visitors will win
2009/01/02|FOTBAL|MUZI|SEVEROIRSKÁ LIGA|2008|2009|BANGOR FC|CLIFTONVILLE|1|1|N|0|2,614|
1,439|3,201|1,368|2,389
2009/01/02|FOTBAL|MUZI|SEVEROIRSKÁ LIGA|2008|2009|LINFIELD BELFAST|BALLYMENA UNITED|1|0|N|
0|1,442|1,044|3,787|2,379|6,402
2009/01/02|FOTBAL|MUZI|SEVEROIRSKÁ LIGA|2008|2009|GLENAVON LURGAN|DUNGANNON SWIFTS|2|2|N|
0|2,393|1,366|3,182|1,437|2,619
2009/01/02|FOTBAL|MUZI|SEVEROIRSKÁ LIGA|2008|2009|CRUSADERS BELFAST|NEWRY TOWN|2|1|N|0|
1,618|1,104|3,474|2,014|4,792
2009/01/02|FOTBAL|MUZI|PŘÁTELSKÉ FOTBAL. ZÁPASY|2009|2009|ZANZIBAR|SOMÁLSKO|2|0|N|0|1,250|
1,010|5,000|3,214|9,000
```

Měny

K *BusinessdemoMoney* je přiložen datový soubor obsahující vývoj kurzů české koruny vůči ostatním měnám za určité relativně krátké období. Soubor je uložen v cestě *businessdemo\MoneyData.txt*.

Text 11 ukazuje první tři řádky z tohoto souboru. První řádek je hlavička, popisující ostatní řádky. Hodnoty jsou na řádcích oddělené pomocí svislítky '|'. První hodnota je datum, ke kterému se řádek vztahuje a další hodnoty pak popisují kurz mezi korunou a příslušnou měnou. O jakou měnu a jaké její množství se jedná je zřejmé z prvního řádku (z hlavičky).

Hlavičkou mohou být i jiné řádky, než pouze první. Hlavičkový řádek se pozná tak, že začíná slovem *Datum*. To se hodí, pokud se mění složení uváděných měn.

Text 11: Ukázka z datového souboru, který obsahuje vývoj kurzů mezi českou korunou a ostatními měnami

```
Datum|1 AUD|1 CAD|1 CHF|1 DKK|1 EUR|1 GBP|100 GRD|100 HUF|100 JPY|1 NOK|1 NZD|1 PLN|1 SEK|
100 SIT|100 SKK|1 USD|1 XDR|1 ATS|100 BEF|1 DEM|100 ESP|1 FIM|1 FRF|1 IEP|1000 ITL|100
LUF|1 NLG|100 P
01.01.1999|18,507|19,486|21,872|4,726|35,226|50,009|10,686|13,895|26,520|3,965|15,894|
8,606|3,712|18,713|81,916|30,166|42,436|2,560|87,323|18,011|21,171|5,925|5,370|44,728|
18,193|87,323|15,985
05.01.1999|18,380|19,393|21,734|4,712|35,100|49,359|10,727|13,947|26,143|3,947|15,756|
8,596|3,691|18,460|81,577|29,758|41,901|2,551|87,011|17,946|21,096|5,903|5,351|44,568|
18,128|87,011|15,928
...
```

Vytváření datových souborů

Automatizované vytváření datových souborů pro potřeby *BusinessDemo* aplikací je v podstatě jednorázová akce – soubory se vygenerují, přibalí se do balíčku na web a tím je hotovo. Generuje se navíc pouze soubor pro *BusinessdemoSport* – finanční data jsou stažena z webu ČNB a nepředpokládá se, že by bylo třeba je znovu generovat.

Pro sporty

Soubor se sportovními daty vytváří soubor `GetSportsDataToFile.pl`, který je umístěn v adresáři `C:\cribber\CreateTextFile`.

Pracuje velmi přímočaře. Stáhne si data z databáze projektu, přeformátuje je a zapíše do výstupního souboru.

Příklad použití:

```
C:\Project\cribber\CreateTextFile>GetSportsDataToFile.pl 2009/01/25 2009/04/25
C:\Project\cribber\CreateTextFile>type SportsData.txt
# Match Date (yyyy/mm/dd)|Sport|Sex|Competition Name|Year From|Year To|Home team Name|
Visitors Team Name|Home Score|Visitor Score|Prolongation|Round|Home will win|Home won't
loose|Draw game|Visitors won't loose|Visitors will win
2009/03/21|BASKETBAL|MUZI|NBA|2008|2009|ATLANTA|DALLAS|95|87|N|0|1,444|1,299|12,950|2,336|
2,850
2009/03/21|HOKEJ|MUZI|NHL|2008|2009|BOSTON|LOS ANGELES|2|3|P|0|1,629|1,182|4,304|2,094|
4,079
2009/03/21|HOKEJ|MUZI|NHL|2008|2009|OTTAWA|MONTREAL|5|4|N|0|2,400|1,482|3,874|1,484|2,405
2009/03/21|HOKEJ|MUZI|NHL|2008|2009|FLORIDA|TORONTO|3|1|N|0|1,803|1,260|4,181|1,864|3,363
...
```

Businessman

Businessman běží na serveru. Na rozdíl od *Businessdemo* je *Businessman* vícevláknový. Každá strategie má několik instancí, jejichž parametry se vzájemně kříží. Počet instancí lze nastavit, viz. dále. *Businessman* použít všechny strategie najednou, ale všechny jejich instance postupně. Každý den spustí všechny strategie a jejich instance právě jednou.

Businessman také na rozdíl od *Businessdemo* spolupracuje s databází.

Businessman je implementován v souboru `businessman\BusinessmanApplication\Program.cs`.

Jak Businessman pracuje

Nejprve načte proměnné do *thread-poolu* z tabulky `T105SystemDials` (strana 65). Do spojového seznamu si vloží všechny žijící strategie (tabulka `T301InvestStrategy` na straně 74). Pak začne strategie spouštět a to tak, že vždy, když je nějaké vlákno v *poolu* volné, využije ho pro výpočet další strategie ze spojového seznamu.

Businessman má svůj konfigurační soubor `BusinessmanApplication.exe.config`, který musí být umístěn ve stejném adresáři, kde je uložen i *Businessman*. V souboru (příkladem je text 12) je třeba nastavit adresář, do kterého se budou ukládat logy a také formát jmen souborů s logy. Dále je nutné nastavit mail server a další informace nutné k automatickému zaslání emailu v případě problému. V neposlední řadě musí soubor obsahovat řetězec pro spojení s databází.

Text 12: Příklad konfiguračního souboru *Businessmana*

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <appSettings>
    <add key="LogDir" value="logs" />
    <add key="LogFile" value="log {0}.txt" />
    <add key="DateTimeFormat" value="yyyy-MM-dd hh_mm_ss" />

    <add key="MailFrom" value="soft.projekt@seznam.cz"/>
    <add key="MailTo" value="petsvec@tiscali.cz"/>
    <add key="MailSubject" value="MoneyMaker - Businessman - ERROR"/>
    <add key="SmtpServer" value="192.168.80.2"/>
    <add key="SmtpSecure" value="false"/>
    <add key="SmtpUserName" value="-"/>
    <add key="SmtpPwd" value="-"/>
  </appSettings>
  <connectionStrings>
    <add name="ConnectionString" connectionString="Data source=Microsoft SQL
Server;server=MoneyMaker;Integrated Security=SSPI;database=MMDB" />
  </connectionStrings>
</configuration>
```

Businessman v průběhu své práce vytváří logy. Formát logů je intuitivní, příklad je v textu 13. V log souborech je obsažena informace o počtu strategiích ve frontě, aktuální spotřebě paměti *Businessmana* a ke každé strategii počet jejích spuštěných instancí.

Pokud nastala v instanci chyba (instance způsobila vznik výjimky), je do logů zapsána pouze výjimka, která byla způsobena první instancí – vzhledem k tomu, že instance jsou až na hodnoty proměnných identické, lze předpokládat, že ostatní instance způsobily výjimku stejného typu (pokud vůbec nějakou způsobily). Při každém selhání instance jsou instance takzvaně napomenuty – pokud selhávají opakovaně a napomenutí se jim kupí, je celá strategie ukončena.

Protože zapisujeme logy trochu netradičně do databáze, nebylo možné použít již vyvinuté systémy pro logování (např. *log4NET*), ale bylo nutné implementovat vlastní logovací systém.

Text 13: Ukázka logů vyprodukovaných *Businessmanem*

```
=== Start === 2009-02-19 03_00_00
Strategies in queue: 4
Memory usage: 913 592
Strategies in queue: 3
Memory usage: 914 688
Strategies in queue: 2
Memory usage: 915 536
Strategies in queue: 1
Memory usage: 916 032
Instances of strategy (2): 100
Instances of strategy (1): 100
Instances of strategy (8): 100
Instances of strategy (10): 100
=== End === 2009-02-19 03_01_00
```

Businessman načítá všechna potřebná data z databáze a stejně tak je tam i ukládá.

Struktura pluginu

Plugin musí být knihovna *.NET* (*.NET Assembly*). Struktura pluginu je povinně takováto:

```
using System;
using System.Collections.Generic;
using System.Text;
using BusinessmanCommon;

namespace BusinessmanPlugin
{
    public class Plugin : PluginBase
    {
        public Plugin():base() { }
        public override void run(PackBase pack) { }
    }
}
```

Plugin pro svoji kompilaci potřebuje knihovnu `BusinessCommon.dll`* obsahující šablony pro plugin a datové položky `MoneyPack`, `SportsPack` atd.

Třída PluginBase

Třída `PluginBase` je abstraktní třída, ze které musí být odvozena třída `Plugin`. Třída je implementována v souboru `businessman\BusinessmanCommon\PluginBase.cs`.

Podstatnou funkcí této třídy je funkce `setBasicData`. Jejím úkolem je logicky svázat knihovnu pluginu s číslem odpovídající strategie a číslem odpovídající instance.

Dále obsahuje proměnné:

<code>InvestInstanceId</code>	ID dané instance
<code>InvestStrategyId</code>	ID dané strategie
<code>Title</code>	Název strategie, který si volí uživatel

Další funkcí je funkce `log`, která zapisuje logy do databáze – tabulka `T310InvestInstanceLog` na straně 81.

Funkce `loadVariables` spouští všechny instance dané strategie – z `dll` knihovny vytvoří *assembly*, vytvoří příslušný `MoneyPack/SportPack` a postupně spouští všechny instance: logicky sváže objekt pluginu s jeho instancí, spojí datové položky a objekt pluginu, načte z databáze hodnoty proměnných. Potom instanci spustí. Po doběhnutí uloží stav (hodnoty proměnných) do databáze.

* V celém projektu jsou po zkompileování tři různé *BusinessCommon.dll* soubory:

1. obsahuje pouze definice objektů pro finanční pluginy,
2. obsahuje pouze definice objektů pro sportovní pluginy,
3. obsahuje vše (finanční i sportovní).

Třetí ze souborů je nahrán pouze na serveru a uživateli se do rukou vůbec nedostane. Předchozí dva jsou přibaleny k příslušnému *BusinessDemo* souboru. Při kompilaci lze využít libovolný z těchto souborů, pro běh *BusinessDemo* je ale třeba ten, který k němu přísluší.

Z databáze se načítají jednak hodnoty proměnných druhu `property` a `field`, které jsou určeny ke křížení, ale také hodnoty proměnných, které se nekříží, ale jsou serializované – toto opatření je příjemné pro uživatele – může počítat s tím, že co si do proměnných uloží, tam bude i při dalším spuštění. Tyto proměnné se ukládají do tabulek `T304InvestInstanceVariables` a `T305InvestInstanceSerializedVariables` (strana 79 a 80).

Poslední důležitou funkcí je funkce `runWithPermissions` – ta bezpečně* zavolá funkci `run`. Tuto funkci volá právě *Businessman* pro spočítání tipů na další den (*BusinessDemo* aplikace volají přímo funkci `run`).

Tvorba pluginu a další informace jsou podrobně popsány v uživatelském návodu.

Moderátor

Moderátor (Obrázek 8) umožňuje jednoduše a pohodlně editovat data v databázi. Předpokládá se, že ho bude používat správce pro editaci uživatelů, skupin, strategií a dalších

Id	Group name	Access rights	Note
1	ADMINISTRATOR	RWRWRWRWRW	Spravce systemu
2	SUPERVISOR	--R--R-RW	Prohlíží strategie
3	USER	---RW--R-	Zadává své strategie

Obrázek 8: Moderátor

* více o bezpečnosti v kapitole *Bezpečnost* na straně 116

dat uložených v databázi.

Celá tato aplikace je napsána v *Delphi*.

Princip práce Moderátora

Zdrojové kódy *Moderátora* jsou uloženy v adresáři `moderator`.

Po té, co je *Moderátor* spuštěn, jako první načte svůj konfigurační soubor (detaily níže).

Následuje autentizace uživatele, viz. strana 39. Pokud je uživatel autentizován, vytvoří se hlavní formulář aplikace (strana 44). Pokud je ten uzavřen, aplikace je ukončena.

Jednotlivé komponenty Moderátora

Konfigurační soubor

Soubor: `moderator\App.config`

Soubor je zpracován pomocí funkcí ze souboru `moderator\UConfig.pas` a důležité informace jsou uloženy do globálních proměnných. Jméno konfiguračního souboru je uloženo v proměnné `ConfigFileName` ve zmíněném souboru.

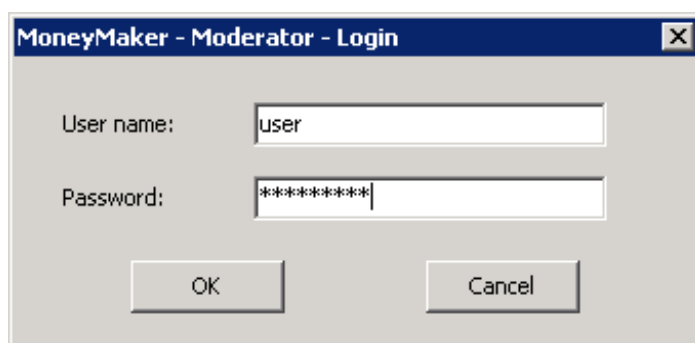
V konfiguračním souboru je nutné nastavit přístup k databázi a kontaktní emailové informace. Email je zaslán na adresu uživatele z adresy zadané v konfiguračním souboru vždy, když se stav uživatele změní z hodnoty `New User` na hodnotu `Enabled user`, tedy když uživatel získá právo nahrávat své pluginy do systému *MoneyMaker*. Více v povídání o tabulce `T101Users` na straně 60.

Ukázkový konfigurační soubor je v textu 14.

Text 14: Příklad konfiguračního souboru Moderátora

```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <configuration>
3   <appSettings>
4     <add key="MailFrom" value="soft.projekt@seznam.cz"/>
5     <add key="AccountActivatedMailSubject" value="MoneyMaker - Account activated"/>
6     <add key="AccountActivatedMailBody" value="Hello,\n\nYour MoneyMaker account has
been activated.\nYou can now use your account.\n\nKind Regards,\n\nMoneyMaker team\n"/>
7     <add key="SmtpServer" value="localhost"/>
8     <add key="SmtpSecure" value="false"/>
9     <add key="SmtpUserName" value="-"/>
10    <add key="SmtpPwd" value="-"/>
11  </appSettings>
12  <connectionStrings>
13    <add name="ConnectionString" connectionString="Driver={SQL Native
Client};Server=localhost\SQLEXPRESS;Database=MMDB;Trusted_Connection=yes;" />
14  </connectionStrings>
15 </configuration>
```

Autentizace uživatele



Obrázek 9: Autentizace uživatele

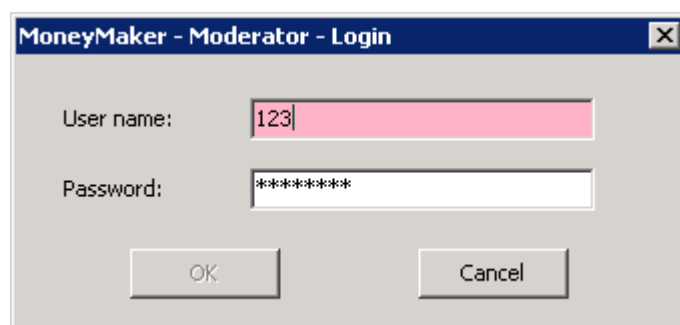
Tento proces je zajišťován kódem ze souboru `moderator\ULogin.pas`.

Při tvorbě formuláře sloužícího k autentizaci je inicializováno spojení s databází procedurou `DBInit`.

Autentizace (funkce `UserLogin` v souboru `moderator\UUserInfo.pas`) využívá tabulku uživatelů z databáze (`T101Users`). Zahešuje (viz. odstavec na straně 43 s nadpisem *DBMakePassword*) a ověří uživatelské jméno a heslo, pokud jsou v databázi, načte do globálních proměnných další informace o uživateli a jeho skupině. Ostatní funkce v souboru `moderator\UUserInfo.pas` nám dávají informace o aktuálním uživateli a jeho skupině.

Pokud proběhne autentizace úspěšně, zobrazí se uživateli hlavní formulář. Pokud se nepovede, vypíše se chybové hlášení a dá uživateli další možnost autentizovat se.

Validace vstupu



Obrázek 10: Neplatný vstup je vyznačen růžovým pozadím (příliš krátké uživatelské jméno)

Grafická komponenta pro *Delphi* upravená pro naše účely.

Soubory: `moderator\UEditValidating.pas`, `moderator\RegExpr.pas`

Soubor `UEditValidating.pas` rozšiřuje standardní komponentu *Delphi* pro validaci editačních polí. Využívá regulární výrazy, které jsou implementovány v souboru `RegExpr.pas`, který napsal Andrey V. Sorokin a je volně dostupný na Internetu.

Účelem tohoto rozšíření je v reálném čase reagovat na vstup uživatele a změnou barvy pozadí editačního pole ho informovat, zda je jeho vstup validní, či nikoliv. V komponentě *Moderátora* je použita růžová barva pro upozornění na nevalidní vstup, viz obrázek 10.

Jak má vstup vypadat určuje programátor na dvou úrovních:

1. regulárním výrazem
2. validační funkcí

Aby byl vstup validní, musí být validní na obou úrovních. Ve výchozím stavu je regulární výraz prázdný (validní je tedy libovolný vstup) a validační funkce je nastavena na funkci `ValidationRoutineTrue`, která vrací vždy `True` a tedy vstup je vždy označen za validní.

Po každé, když uživatel vstup změní, zavolají se obě validační metody (procedura `MaskTextChanged`). Pokud se validita vstupu změnila, je zavolána funkce uložená v `FvalidationAction`, kterou si programátor může libovolně nastavit – například může změnit platnost jiného pole atd. Příkladem budiž opět obrázek 10 – uživatelské jméno není zadáno validně a proto se tlačítko OK zakázalo. V okamžiku, kdy bude mít uživatelské jméno dostatečný počet znaků, se tlačítko zase povolí.

Programátor má samozřejmě možnost obě metody nastavit dle svých potřeb pomocí procedur `ValidationMaskSet` a `ValidationRoutineSet`.

Nastavením příslušných konstant lze měnit barvy editačních polí podle výsledku validace:

<code>VALIDATING_COLOR_NOERROR</code>	Barva pozadí při validním vstupu
<code>VALIDATING_COLOR_ERROR</code>	Barva pozadí při chybném vstupu
<code>VALIDATING_COLOR_DISABLED</code>	Barva pozadí, pokud je editační pole zakázané

Barvy validního a invalidního vstupu lze měnit i za běhu pomocí procedur `ValidationColorErrorSet`, `ValidationColorNoErrorSet`.

Mezi konstantami se nachází i pole `Accents`, které je použito pro převod textu s diakritikou na text bez diakritiky (procedura `RemoveAccents`). Před porovnáváním vstupu a regulárního výrazu je diakritika odstraněna, protože použitá implementace regulárních výrazů s ní neumí pracovat.

Obrázek 11: Filtr

Filtr dat

Filtr je modul, který pomáhá získávat z databáze data podle požadavků uživatele, které jsou graficky zadané a zároveň mu zobrazuje pouze data, na která má práva pro čtení. *Filtr* informace z databáze pouze pomáhá získávat, nemění je. Součástí *filtru* je tvorba grafického rozhraní. Příklad filtru je na obrázku 11.

Zdrojové kódy se nacházejí v souboru `moderator\UFilter.pas`.

Filtr implementuje tři typy filtrace:

1. Seznam

Je možné zadat seznam povolených hodnot, které se uživateli zobrazí jako tzv. *Combobox* (na obrázku např. `Kind`, `Language`, `State`). Uživatel si může vybrat z možností `is` a `is not`, nebo `without limitation`.

2. DateTime

Vhodné pro určení data. Na obrázku je tohoto typu položka `Insert time`. Uživatel může čas specifikovat buď jako interval od zvoleného data do teď, od zvoleného data do minulosti, nebo ohraničit interval z obou stran. Také může zadat `anytime`, tedy zobrazit všechny výsledky nezávisle na této položce.

3. String

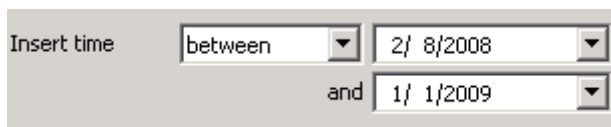
Tohoto typu jsou na obrázku políčka `User name` a `Title`. Uživatel může zadat libovolný řetězec a zvolit si, zda se má vyhledat řetězec, který jím začíná, končí, obsahuje ho, neobsahuje ho, nebo je zcela stejný. Samozřejmě může opět nastavit pole na `without limitation`, aby viděl všechny možné výsledky.

Hlavní třídou *filtru* je `TSearchFilter`. Součástí této třídy je proměnná `FList`, seznam položek *filtru* (položkou *filtru* se myslí jeden jeho „řádek“). Funkce `AddList`, `AddString` a `AddDateTime` do tohoto listu přidávají položky. Funkce `Remove` je odebírání. Funkce `ResetFilter` vynuluje hodnoty všech položek na výchozí hodnoty.

Po vyplnění formuláře uživatelem je ze zadaných hodnot vytvořen SQL dotaz. K tomu slouží funkce `GetSQLQuery`. Tato funkce vytvoří tu část dotazu za klíčovým slovem `WHERE`. Postupně bere každou položku *filtru*, sestrojí podle ní příslušnou část dotazu a tyto pospojuje do jednoho celku. Každý typ položky vyžaduje jiné zpracování, takže funkce `GetSQLQuery` používá funkci `GetFilterQuery`, která je definována pro každý typ položky zvlášť.

Každý typ položky je definován svou vlastní třídou, konkrétně jsou nadefinovány třídy `TSearchFilterItemList`, `TSearchFilterItemString` a `TSearchFilterItemDateTime`. Všechny mají společného předka, třídu `TSearchFilterItem`, která má některé funkce definovány jako čistě virtuální (např. právě funkci `GetFilterQuery`).

Proměnná `FType` v třídě uchovává informaci, jakého je položka typu – zda je `List`, `String`, nebo `DateTime`.



Obrázek 12

Proměnná `FName` určuje, jakého sloupce v databázové tabulce se položka týká. Musí zde být uložen stejný řetězec, který je použit pro název sloupce v tabulce (např. `UserID`).

Mezi další významné proměnné třídy patří souřadnice, které uvádějí, kde bude položka ve formuláři umístěna.

Tři zmíněné třídy pro jednotlivé typy položek (`List`, `String` a `DateTime`) se od sebe liší požadavky, které na ně klademe, a tedy také svým obsahem. Některé proměnné mají ale společné. Mezi ty významné patří `FLabel`, která obsahuje popis položky, který ze zobrazí uživateli. Na obrázku 12 je `FLabel` nastavena na `Insert time`. Za zmínku stojí ještě funkce `PositionUpdate`, také součástí všech tří tříd. Tato funkce definuje vzájemnou polohu prvků položky ve formuláři. Jak daleko jsou od sebe, v jakém pořadí atd.

Třída `TSearchFilterItemString` obsahuje seznam pro výběr uživatele `FComboBox`, znázorňující vyžadovaný vztah. Na obrázku 11 je to ten box, který obsahuje slovo `equals`. Editační pole `FEdit` slouží k vyplnění libovolným řetězcem, na obrázku 11 je to pole obsahující slovo `Pepicek`. Položka `FComboBoxOption` obsahuje aktuálně vybranou možnost ze seznamu `FComboBox`.

Třída `TSearchFilterItemList` je předchozí třídě podobná, obsahuje seznam pro výběr uživatele `FComboBoxVerb`, znázorňující vyžadovaný vztah. Na obrázku 11 je to ten box, který obsahuje slovo `is`. Editační pole je nahrazeno opět seznamem pro výběr `FComboBoxValue`, na obrázku 11 je to např. pole obsahující slovo `MONEY`. Položka `FComboBoxVerbOption` obsahuje aktuálně vybranou možnost ze seznamu `FComboBoxVerb`. Možné hodnoty zobrazené v boxu `FComboBoxValue` jsou uloženy v seznamu `FValues`. Do tohoto seznamu jsou přidávány funkcí `AddListItem`.

Třída `TSearchFilterItemDateTime` je opět velmi podobná, ale má tři *Comboboxy* (viz. obrázek 12) a navíc má boolovskou proměnnou `FTwoLiner`, která určuje, zda bude položka zobrazena na dvou řádcích, jako na obrázku 12, nebo bude celá v jednom řádku.

Lokalizace

Pro potřeby lokalizace je v aplikaci *Moderátor* pouze jediný soubor, který obsahuje všechny uživateli zobrazované řetězce textu, které nejsou součástí samotného *GUI*. Tak je lze snadno přeložit do jiného jazyka, aniž by musely být složitě vyhledávány.

Jedná se o soubor `moderator\UStrings.pas`.

Typy

Často je potřeba převést data mezi různými typy. Funkce pro to určené jsou implementovány v souboru `moderator\UTypes.pas`. Příkladem použití takových funkcí je převod na řádku 5 v textu 15 na straně 43. Soubor obsahuje rovněž definice typů specifických pro *Moderátora*.

Práce s databází

Obecný přehled

Tyto funkce lze ještě dále rozdělit na tři kategorie:

- (1) režijní funkce (inicializace databáze, výpis chybových hlášení atd.)

(2) funkce, které pracují přímo s dotazy do databáze

(3) funkce, které volají předchozí typ funkcí a přidávají další funkčnost – ty jsou volané funkcemi z dalších modulů

Jako příklad typické funkce, která přímo volá funkce databáze (2) si ukažme funkci DBT105Update (Text 15). Funkce si připraví dotaz pomocí funkce DBQueryPrepare (řádek 3) a pak dosadí konkrétní hodnoty parametrů (řádky 5 až 9). Tím zajistí, že vložené řetězce nebudou obsahovat „neescapované“ znaky klíčové pro SQL a vykoná dotaz (řádek 11). Vrací hodnotu, kterou vrátila funkce databáze.

Text 15: Ukázková funkce volající funkci databáze

```
1 function DBT105Update(AValId:Integer;AName,AType,AValue,ANote:string):Integer;
2 begin
3   DBQueryPrepare(DBQuery,'T105Update',':id, :name, :type, :value, :note');
4
5   DBQuery.Parameters.ParamByName('id').Value:=AValId;
6   DBQuery.Parameters.ParamByName('name').Value:=AName;
7   DBQuery.Parameters.ParamByName('type').Value:=AType;
8   DBQuery.Parameters.ParamByName('value').Value:=AValue;
9   DBQuery.Parameters.ParamByName('note').Value:=ANote;
10
11  Result:=DBQueryFuncCall(DBQuery);
12 end;
```

Podívejme s nyní, co se děje s návratovou hodnotou dále. Příkladem volání této funkce a zpracování její návratové hodnoty je funkce DBSystemDialsUpdate (funkce typu (3)), jejíž část obsahuje text 16.

Text 16: Zpracování návratové hodnoty

```
LDBRes:=DBT105Update(AValId,AName,LTypeStr,AValue,ANote);
if LDBRes<0 then LDBRes:=LDBRes-90
DBErrorHandler(LDBRes);
```

Funkce DBT105Update je zavolána a pokud je její návratová hodnota chybová, tedy menší, než nula, je od této hodnoty odečtena ještě konstanta. Volající funkce se liší velikostí konstanty, kterou v případě chyby odečítají, tedy je možné z chybové hodnoty zjistit jak její smysl, tak která funkce ji zpracovávala. Pokud dostanu např. hodnotu -72, je zřejmé, že chybová hodnota byla -2 (chybové hodnoty jsou pouze v intervalu od -1 do -9) a zpracovala ji funkce, která odečítá hodnotu 70. Všechny funkce typu (3) volají funkce typu (2) a výsledek zpracovávají obdobně – volají na něj funkci DBErrorHandler.

Funkce DBErrorHandler má za úkol vypsát chybové hlášení, pokud k nějaké chybě došlo. Obsahuje v sobě tabulku spojující číslo chyby a řetězec, který chybu popisuje uživateli (řetězce jsou brány ze souboru UStrings.pas. viz. téma Lokalizace).

DBMakePassword

Důležitou funkcí databázové části je funkce DBMakePassword, která se používá při autentizaci uživatele. Dostane jméno a heslo a zahešuje je. Výsledný řetězec je pak možno porovnat se záznamy v databázi. Tato funkce využívá hešovací funkci StrSHA256, která je umístěna v souboru SHA.pas. Tento soubor je volně dostupný na Internetu a jeho autor je

neznámý.

Hlavní formulář aplikace

Hlavní formulář aplikace ukazuje obrázek 8 na straně 37. V každé záložce formuláře, až na výjimky, jsou obsaženy tyto prvky:

- *filtr* (na obrázku 8 vlevo nahoře)
- sekce *new / edit* (na obrázku 8 vpravo nahoře)
- tlačítka (na obrázku 8 zcela vpravo nahoře)
- tabulka výsledků, tzv. *grid* (na obrázku 8 dole)

Formulář je implementován v souboru `moderator\UMain.pas`.

TGridData

TGridData tvoří tabulku výsledků dotazu do databáze, je vidět ve spodní části obrázku 8. TGridData je typu `record` a obsahuje mimo jiné:

- data nutná ke spojení s komponentou `grid`
- *handler* události typu změna výběru v tabulce
- nadpisy sloupečků a jejich implicitní šířku
- spojení s příslušným filtrem
- části SQL dotazů, jejichž výsledky zobrazují
- spojení s tlačítkem, kterým se filter spustí a aktualizuje data v `gridu`

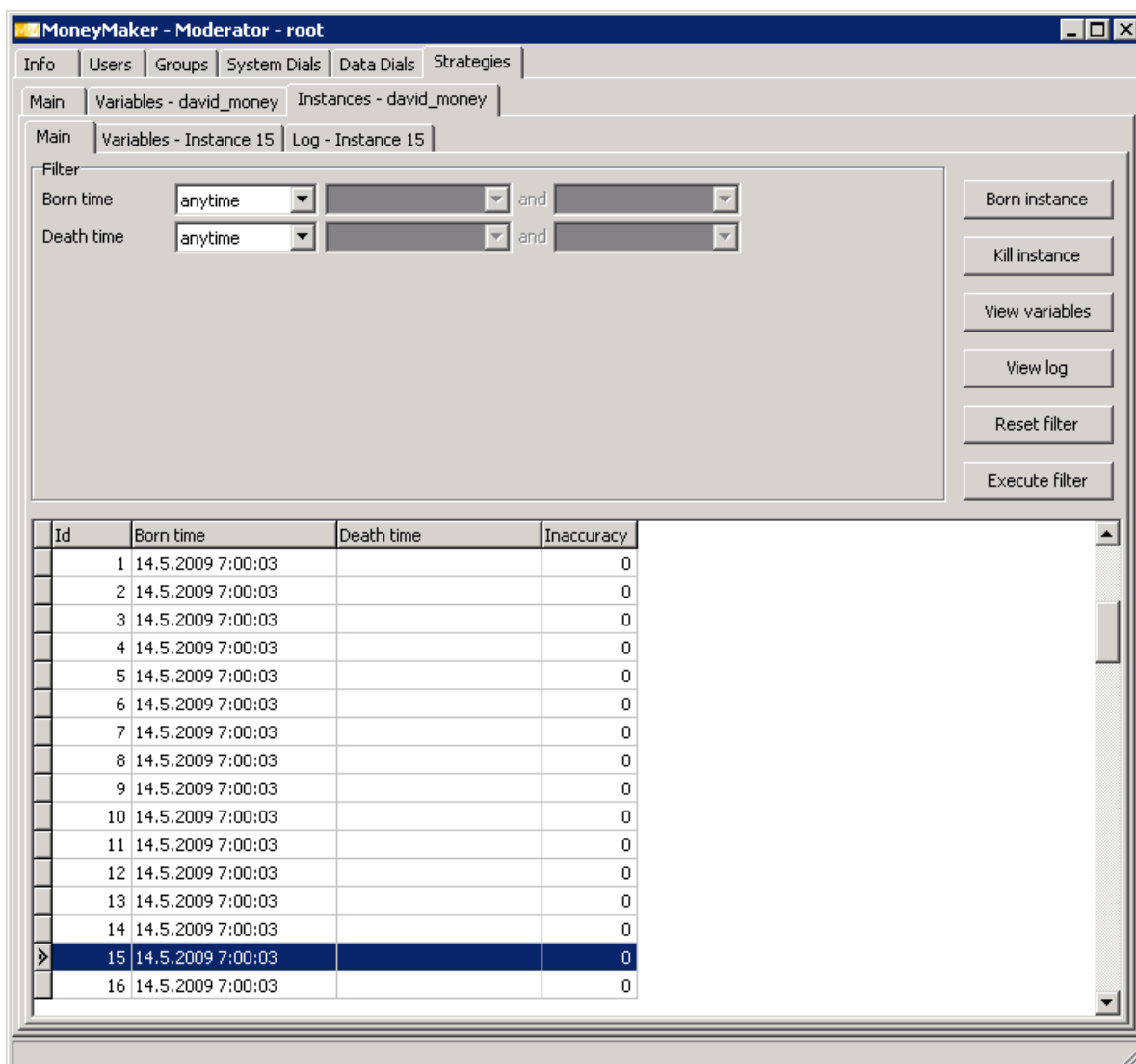
Soubor obsahuje dvě zajímavé konstanty:

- **GridMaxRecords**: maximální počet výsledků, které se do `gridu` vypíší. Aktuálně nastaven na konstantu 100. Myšlenka tohoto opatření je taková, že pokud mému dotazu odpovídá více, než 100 výsledků, je pravděpodobně špatně zformulovaný – je třeba ho více specifikovat.
- **GridCount**: konstanta uchováající v sobě celkový počet `gridů` v aplikaci. Každý `grid` má své pořadové číslo. Čísla jsou definována následujícím způsobem:

Text 17: Definice čísel gridů

```
TagUsers           = 0;
TagGroups          = 1;
TagSystemDials     = 2;
TagDataDials       = 3;
TagStrategies      = 4;
TagStrategiesMain  = TagStrategies;
TagStrategyVariables = 5;
TagStrategyInstances = 6;
TagInstancesMain   = TagStrategyInstances;
TagInstanceVariables = 7;
TagInstanceVariablesMain = TagInstanceVariables;
TagInstanceVariableHistory = 8;
```

| TagInstanceLog = 9;



Obrázek 13: pohled pro pochopení očíslování gridů

Podívejme se znovu na obrázek 13 a srovnajme ho s textem 17. Platí, že co záložka, to jeden grid s několika výjimkami: záložka Info nemá grid žádný, záložka Strategies má gridů hned několik. Strategies má několik pod-záložek a každá pod-záložka má svůj grid, podobně pod-záložka Instances má další pod-pod-záložky s gridy. Pro všechny úrovně platí, že záložka, která má pod-záložku, má jako svůj grid uveden grid hlavní pod-záložky.

Každý grid má v souboru určené své vlastnosti, jako implicitní šířky sloupců a jejich nadpisy. Každý grid má také svůj filtr.

Vytvoření formuláře

Při vytvoření formuláře se provede inicializace, vytvoří se filtry, pro každý grid se vytvoří „dotazovatel“ do databáze, vytvoří se sloupečky gridů a asociují se taby a gridy. Vyplní se záznam TGridData včetně částí SQL dotazů. Názvy sloupců tabulek pochopitelně musí odpovídat názvům používaným ve filtru, jinak by dotaz nedával smysl (jednotlivé části dotazu

budou pospojovány, jak již bylo zmíněno). Dále je nastaveno implicitní zobrazení – je vidět záložka Info.

Když je formulář poprvé zobrazen, získá informace o přihlášeném uživateli – jsou uloženy v proměnných `UserInfo` a `GroupInfo` – je nutné je mít uložené kvůli právům. Potom je „uměle“ kliknuto na tlačítko spuštění filtru, aby se grid naplnil daty. Podle toho, jaká má uživatel práva se pravděpodobně zakáží některá pole a tlačítka formuláře. Na konec inicializace se ještě nastaví validační masky a procedury.

Správa uživatelů a skupin

Pro editaci uživatele slouží formulář, které se zobrazí v samostatném okně po kliknutí na tlačítko Add user – detaily níže. Uživatel ale nemusí mít práva pro editaci nebo i jen pouhé čtení informací o ostatních uživateli a tak mu zůstane záložka Users zapovězena. Vždy ale musí být schopen editovat informace o sobě. K tomu je určeno tlačítko Change my user information v záložce Info, které zobrazí stejný formulář, jako tlačítko Add user v záložce Users, ale některé možnosti v něm budou zakázané. Konkrétně uživatel nebude moci měnit položky Nick, Active a Group.

Zmiňovaný formulář je implementován souborem `moderator\UUsersAdd.pas`. Implementuje správu uživatelů a rozesílání mailů – v případě, že se stav uživatele změní na active, je mu zaslán informační email (data pro tuto operaci jsou načítána z konfiguračního souboru).

Tento soubor je psaný přímočaře, obsahuje pouze jeden malý „trik“: v proměnné `Tag` je uchovávána buď nula, nebo, pokud je to možné, ukazatel na datovou strukturu obsahující informace o aktuálně editovaném uživateli. Díky tomu se políčka formuláře předvyplňují daty z databáze.

Uživatelská data jsou ukládána do tabulek `T101Users` a `T102UserAdditionInformation` (stránky 60 a 61).

Formulář samozřejmě také využívá validační modul.

Poloautomatizace

V souboru se nachází relativně hodně funkcí, které jsou společné pro všechny gridy, i když se k nim chovají mírně rozdílně. Takové funkce jsou např. `TitleClick` (pokud je kliknuto na nadpis sloupce tabulky, tak ho abecedně seřadí), `CellClick` (při kliknutí na řádek gridu) nebo `FilterExecuteClick`.

Genesis

Genesis je součástí serveru. Když uživatel nahraje na server svůj plugin (.NET Assembly), genesis ho zanalyzuje. Nalezne v něm všechny proměnné, které by se měly křížit (veřejné proměnné základních typů deklarované přímo ve třídě `Plugin`).

Genesis vyžaduje při spouštění dva parametry:

- jméno *dll* souboru, který má analyzovat (knihovna poskytnutá uživatelem)
- ID strategie (aby bylo možné logicky spojit proměnné, plugin a konkrétní strategii, které patří k sobě)

Např.:

```
| genesis.exe strategy.dll 18
```

Je možné *genesis* zadat jeden parametr `-NOCONSOLE`, který způsobí, že nebude nic vypisovat na konzoli během svého běhu, ale pouze do logovacího souboru.

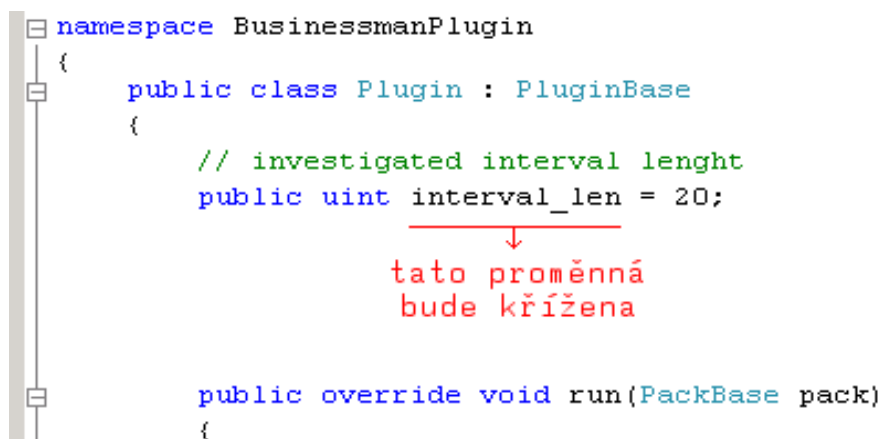
Genesis je spouštěna pouze modulem *Occupant* při nahrání nové strategie. O výsledku operace informuje *Occupant* pomocí návratové hodnoty – je volána jako zcela samostatný program.

Princip práce programu

Prvně samozřejmě zpracuje parametry. Pak zinicizuje spojení s databází a smaže stará data uložená v databázi týkající se zpracovávané strategie – budou přepsána novými daty.

Strategie dále prochází plugin a pro každou veřejnou property, nebo field, které jsou zapisovatelné:

1. zjistí aktuální hodnotu
2. zjistí jméno
3. zjistí typ
4. pokud se jedná o atomický typ*, uloží hodnoty do databáze (tabulka `T302InvestStrategyVariableRange` na straně 76)



```
namespace BusinessmanPlugin
{
    public class Plugin : PluginBase
    {
        // investigated interval lenght
        public uint interval_len = 20;

        public override void run(PackBase pack)
        {
```

tato proměnná
bude křížena

Obrázek 14: křížení proměnných

Pokud je nalezena hodnota, která se nemůže křížit, ale přesto je veřejná a serializovatelná, je zaznamenáno varování, které ve výsledku probublá k uživateli na web.

Totéž, co dělala *genesis* pro každou property, udělá pro každý field.

* Atomické typy jsou *boolean*, *char*, *integer*, *unsigned integer*, *float*, *double*, *decimal*, *string* a *long*

Celá akce zpracování proměnných je umístěna *try-catch* bloku, tedy pokud nastane problém, je objeven a ošetřen zasláním emailu – viz. konfigurační soubor *genesis*.

Druhy proměnných vzhledem k možnostem křížení

Některé hodnoty se kříží, jiné se jen ukládají a znovu načítají při dalším spuštění pluginu a některé ani jedno, ani druhé.

Aby se hodnota proměnné ukládala a načítala, musí být proměnná `public` a `serializable` a navíc musí být definována právě ve třídě pluginu (mimo tělo funkce `run`), ale v těle třídy. Pokud je taková proměnná zároveň jednoduchého typu (viz. obrázek 13), ukládá se do tabulky `T304InvestInstanceVariables` (strana 79), pokud je složeného typu, ukládá se do tabulky `T305InvestInstanceSerializedVariables` (strana 80). Kříží se pouze jednoduché proměnné – u nich si uživatel vybere, zda je chce křížit, nebo ne, na webu, nebo pomocí *Moderátora*. Ostatní proměnné nelze křížit. Zda se budou proměnné křížit, nebo ne, lze zvolit pro celou strategii. Tato volba to je nadřazená jednotlivým volbám o konkrétních proměnných.

Konfigurační soubor

Text 18: Ukázkový konfigurační soubor genesis

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <appSettings>
    <!-- logovani -->
    <add key="LogDir" value="logs" />
    <add key="LogFile" value="log {0}.txt" />
    <add key="DateTimeFormat" value="yyyy-MM-dd hh_mm_ss" />

    <!-- zasilani informacniho emailu v pripade chyby -->
    <add key="MailFrom" value="soft.projekt@seznam.cz"/>
    <add key="MailTo" value="petsvec@tiscali.cz"/>

    <add key="MailSubject" value="MoneyMaker - Genesis - ERROR"/>
    <add key="SmtpServer" value="192.168.80.2"/>
    <add key="SmtpSecure" value="false"/>
    <add key="SmtpUserName" value="-"/>
    <add key="SmtpPwd" value="-"/>
  </appSettings>
  <connectionStrings>
    <!-- pristup do databaze -->
    <add name="ConnectionString" connectionString="Data source=Microsoft SQL
Server;server=MoneyMaker;Integrated Security=SSPI;database=MMDB" />
  </connectionStrings>
</configuration>
```

Soubor musí být uložen ve stejném adresáři jako spuštěný binární soubor *genesis* a pojmenován *genesis.exe.config*.

Logy

Genesis zaznamenává svůj běh do logů. Jména souborů s logy jsou nastavitelná pomocí konfiguračního souboru, jejich formát je intuitivní. Log obsahuje datum a čas akce, typy, jména a hodnoty proměnných. Ukázka takového logu je v textu 19.

Text 19: Ukázkový log genesis

```
=== Start === 2009-02-09 05_16_11
Double D2 = 45
UInt32 days = 5
String txt = ''dfsd'sdf'
Single f1 = 12
Single f2 = 12.3
Double d1 = 12
Decimal dec1 = 123
Decimal dec2 = 123.213
=== End === 2009-02-09 05_16_12
```

Occupant

Webové rozhraní frameworku je nazvané *Occupant* a lze ho nalézt ve stejnojmenném adresáři. Popišme si nejprve stručně, jaké má části a kde se nacházejí.

Adresář	Popis obsahu
conf	Nastavení (nastavuje se pouze přístup k databázi)
inc	Soubor skriptů využívaných dalšími stránkami
style, images	Design (rozvržení objektů na stránce, obrázky)
sports, strategies, kořenový adresář	Každý <i>.php</i> soubor z těchto adresářů odpovídá nějaké konkrétní stránce, která je zobrazována uživateli

Obecná struktura průběhu načítání stránky

Princip je u všech stránek téměř stejný a zhruba odpovídá následujícímu schématu:

1. Stránka nejprve načte soubor *.hroot.php* ve svém aktuálním adresáři. Tento soubor je obsažen v každém adresáři přístupném přes web. Slouží pro nadefinování důležitých konstant a provedení operací, které se provádějí na každé stránce. Aktuálně je v něm definovaná konstanta *ROOT*, která v sobě obsahuje relativní cestu do kořenového adresáře. Když je známa cesta ke kořenovému adresáři, je načten soubor */inc/page.php* obsahující základní kostru stránky. Nakonec se vytvoří instance objektu stránky (objekt *Page*). Tímto role souboru *.hroot.php* končí.
2. Jsou nastaveny nadpisy stránky, případně načteny další *JavaScript* skripty.

3. Pokud se mají zpracovat vstupy od uživatele (metodami POST, nebo GET), zpracují se v této fázi. Na zabezpečených stránkách se v tuto chvíli také kontroluje autenticita a oprávnění uživatele.
4. Pokud je třeba, načtou se další pomocné *PHP* skripty.
5. Je zavolána funkce `page->head()`, která vypíše *HTML* kód obsahující hlavičku stránky.
6. Vypíší se hlavní data stránky.
7. Je zavolána funkce `page->foot()`, která vypíše *HTML* kód obsahující ukončení stránky.

Některé části podrobněji

Třída Page

Vypisuje hlavičku a patičku stránky, hlavní menu na stránce i menu pro výběr jazyka – prvky společné všem stránkám. Třída je nadefinována v souboru `inc\page.php`.

Lokalizace

Stránky podporují dva jazyky – češtinu a angličtinu a toto nastavení lze jednoduše rozšířit i pro další jazyky. Všechny funkce zajišťující vícejazyčnost webu se nacházejí v souboru `inc\lang.php`.

Nejpoužívanější funkcí je funkce pojmenovaná `_` (podtržítko). Tato funkce přeloží zadaný řetězec do jazyka aktuálně zvoleného uživatelem. Pro překlad používá tabulku výrazů, která je uložena v souboru `lang_translate_table.php`. Ukázka, jak tabulka vypadá, můžete vidět v textu 20 – jedná se o jednoduché asociativní pole, kde klíč je výraz v jednom jazyce a hodnota je výraz přeložený do druhého jazyka.

Pokud se při „překladu“ narazí na výraz, který v tabulce není a tedy ho přeložit nelze, je zapsán do souboru `untranslated.log`. Tento soubor by měl administrátor stránek pravidelně kontrolovat a případné nepřeložené výrazy obratem přeložit. Tato funkčnost se hodí zvláště v případě rozsáhlejších změn na webu, kdy se nepodaří uhlídat překlad všech výrazů.

Text 20: Ukázka překladové tabulky

```
$lang_translate_table = array(
    "cs" => array(
        "Welcome to MoneyMaker" => "MoneyMaker Vás vítá",
        "Welcome" => "Vítejte",
        "Home" => "Domů",
        "Profile" => "Profil",
        "Charts" => "Grafy",
        ...
    )
);
```

Při prvním přístupu uživatele na stránky, dříve než má možnost kliknout na obrázek vlajky a změnit si tak své nastavení, je třeba zjistit, jaký jazyk použít. K těmto účelům se používá funkce `preferred_language_choose`. Z hlavičky zaslané prohlížečem uživatele je získáno pole

jazyků, které prohlížeč podporuje. Čeští uživatelé zpravidla zasílají v tomto řetězci češtinu i angličtinu, ovšem češtinu s vyšší preferencí. Funkce vybere ten jazyk, který web podporuje (čeština nebo angličtina) a v zaslané hlavičce má nastavenou vyšší preferenci.

Tabulky

Na webu jsou často používané tabulky. Aby bylo možné nastavovat jim parametry globálně a tvořit je jednoduše, vypisují se tabulky pomocnými funkcemi. Funkce najdete v souboru `inc\tables.php`. Kód stránky je s použitím těchto funkcí přehlednější a kratší.

Příklady použití jsou převzaty ze souboru `matches.php`.

PHP kód

```
$header_printer = new SimpleTableHeaderPrinter(array(
    'time' => _('Date'),
    'teams' => _('Teams'),
    'score' => _('Score'),
    '1' => '1',
    '0' => '0',
    '2' => '2',
    '10' => '10',
    '20' => '02',
),
    "<col span='3' align='left' /><col span='5' width='60px' align='right' />");
$row_printer = new SportsTableRowPrinter(true);
$table_printer = new TablePrinter($results, $header_printer, $row_printer);
$table_printer->dump();
```

Vygenerovaný HTML kód

```
<table class="datatable" border="1" cellpadding="0" cellspacing="0"><col align="left"
span="3"><col align="right" span="5" width="60">
<thead>
    <tr class="">
        <th>Datum</th>
        <th>Teams</th>
        <th>Výsledek</th>
        <th>1</th>
        <th>0</th>
        <th>2</th>
        <th>10</th>
        <th>02</th>
    </tr>
</thead>
<tbody>
    <tr class="odd hover">
        <td>2009-03-24 21:00</td>
        <td>TOTTENHAM - MIDDLESBROUGH</td>
        <td>4:0</td>
        <td>1.726</td>
        <td>1.146</td>
        <td>3.411</td>
        <td>2.000</td>
        <td>4.835</td>
    </tr>
    ...
    <tr class="odd">
        <td>2009-02-09 13:45</td>
        <td>ASTON VILLA - WEST BROM</td>
        <td>2:1</td>
        <td>1.424</td>
        <td>1.051</td>
        <td>4.018</td>
        <td>2.641</td>
        <td>7.708</td>
    </tr>
</tbody>
```

Výsledná stránka

</table>

Datum	Teams	Výsledek	1	0	2	10	02
2009-03-24 21:00	TOTTENHAM - MIDDLESBROUGH	4:0	1.726	1.146	3.411	2.000	4.835
2009-03-24 21:00	FULHAM - BLACKBURN	1:2	2.201	1.303	3.194	1.601	3.211
2009-03-24 21:00	FULHAM - HULL	0:1	1.682	1.131	3.456	2.076	5.199
2009-03-24 21:00	WEST HAM - WEST BROM	0:0	1.781	1.165	3.369	1.919	4.460
2009-03-24 21:00	HULL - TOTTENHAM	1:2	3.119	1.584	3.218	1.319	2.234
2009-03-24 21:00	LIVERPOOL - SUNDERLAND	2:0	1.411	1.044	4.010	2.710	8.361

Formuláře

Podobně jako jsou v projektu obsaženy třídy pro vykreslení tabulek, je tu i třída pro vykreslení formuláře. Třída je uložena v souboru `inc\form_renderer.php`. Třída je potomkem třídy `PEAR` (kapitola *Použité veřejně dostupné moduly: PEAR a jQuery* na straně 54).

Příklad znázorňuje použití v souboru `strategies\add.php`. Jedná se o jednoduchý příklad, v dalších souborech lze nalézt i složitější využití.

PHP kód

```
$form = new HTML_QuickForm('firstForm');
$form->addElement('text', 'nick', _('User name'));
$form->addElement('password', 'pwd', _('Password'));
$form->addElement('submit', 'Send', _('Send'));

if ($form->validate()) {
    $r = $db->login($form->exportValue("nick"), $form->exportValue("pwd"));
    switch ($r) {
        case -1:
            $page->head();
            echo "<p>_(_("Internal error happened. Please try again later."))_</p>";
            break;
        case 0:
            $page->head();
            echo "<p>_(_("Wrong Log-in name / password combination or the user account is not activated."))_</p>";
            break;
        case 1:
            header("Location: ".ROOT);
            exit;
    }
} else {
    $page->head();
    // Output the form
    $page->render_form($form);
}
```

Vygenerovaný HTML kód

```
<form id="firstForm" action="/login.php" method="post">

<table class="formtable">
<tbody><tr>
<th><label for="nick">Uživatelské jméno</label></th>
<td><input value="" name="nick" id="nick" type="text"></td>
</tr>
<tr><th><label for="pwd">Heslo</label></th>
<td><input value="" name="pwd" id="pwd" type="password"></td>
</tr>
<tr>
<td></td><td class="formsubmit"><input name="Send" value="Odeslat" type="submit"></td></tr>
</tbody></table>
</form>
```

Uživatelské jméno

Heslo

Relace

Pro každého návštěvníka webu je utvořena jeho *relace*, která obsahuje:

- ID relace
- jazyk, který si uživatel zvolil
- zda je uživatel přihlášen (TRUE/FALSE hodnota)
- uživatelské jméno (pokud je přihlášen; v jeden okamžik může být jedním jménem přihlášeno i více uživatelů a každý má svoji vlastní relaci)
- počet zobrazení webu (tzv. *Hity*)

Tyto údaje jsou uloženy v PHP strukturách `$_SESSION` (nikoli v databázi). ID relace každého návštěvníka je uložena na jeho počítači pomocí *cookies*. ID je také neustále kontrolováno (funkce `validate()`), zda se pojí se stále stejnou IP adresou. Pokud se IP adresa změní, je uživateli přidělena nová relace (a pokud byl přihlášen, musí se přihlásit znovu – funkce `login()`).

Relace jsou implementovány v souboru `session_monitor.php`.

Práce s databází

Práce s databází je rozdělena na dvě vrstvy. První vrstva obsahuje samotné SQL dotazy, druhá je její nadstavbou a již žádné dotazy přímo neobsahuje. První vrstva je implementována v souboru `inc\db_monitor.php`, druhá v souboru `db.php`.

Soubor implementující první (spodní) vrstvu obsahuje také práci s relací. Tedy uvnitř objektu pro práci s databází se nachází instance objektu pro relace, takže se ke všem datům týkajícím se *stavu aplikace* (včetně dat týkajících se relace) přistupuje přes pomyslné databázové rozhraní – a z něj je část *neviditelně* řešena pomocí PHP `$_SESSION`, jak je popsáno v předchozí kapitole.

Horní vrstva načítá konfigurační soubor – aktuálně je v něm nastavitelná jediná hodnota: přístupové údaje k databázi. Soubor se nachází v cestě `conf\db_conf.php` a jeho možný obsah ukazuje text 21 (detaily osvětluje komentář v souboru).

Text 21: Konfigurační soubor – nastavení přístupu k databázi pro web

```
<?php
    /// Defines a connection to the database the site will use
    ///
    /// The format is either a DSN (Data Source Name), that is a string
```

```
/// or an array definig all the properties. Refer to PEAR documentation
/// for more info.
/// Usual format is: phptype://hostspec/database
/// or also: phptype://username:password@protocol+hostspec:110//usr/db_file.db
define ('db_connection_dsn', 'mssql://localhost/MMDB');
```

?>

V této druhé vrstvě je načten kód vrstvy první – pracuje se s třídou `MMDB`, která je implementována právě v první vrstvě. Z této třídy se vytvoří globální instance dostupná pro všechny skripty a její pomocí se přistupuje k databázi.

Tato stránka (druhé vrstvy) je načítána v souboru `page.php`, což zaručuje každé stránce přístup k databázi, viz kapitola *Obecná struktura průběhu načítání stránky* na straně 49.

Použité veřejně dostupné moduly: *PEAR* a *jQuery*

PEAR - PHP Extension and Application Repository

- <http://pear.php.net/>
- řeší některé často se vyskytující problémy při vývoji webu
- *MoneyMaker* projekt využívá tyto části:
 - vykreslování a validace formulářů
 - tvorba grafů
 - *captcha* (při registraci)

jQuery - JavaScript knihovna

- <http://jquery.com/>
- velmi známá a využívaná knihovna
- umí zakrývat rozdíly mezi prohlížeči
- zkracuje zápis operací, které se provádějí často
- *MoneyMaker* projekt využívá tyto části:
 - „vychytávky“ týkající se zobrazované grafiky
 - další validace formulářů (kterou nelze snadno provést pomocí PEAR)

Další užitečné funkce

Nachází se v souboru `utils.php` a obsahují zbylé pomocné funkce, které se nehodí zařadit k žádnému z předchozích témat. Příkladem zde uložených funkcí jsou funkce pro převod data (1. pád: *datum*) mezi jednotlivými formáty – uživatel vyžaduje jiný formát, než SQL databáze, jiný formát vyžadují i některé PHP funkce...

Killguard

Úkolem *Killguard* je probírat se instancemi strategií, vyhodnocovat jejich úspěšnost a podle toho upravovat jejich počet – ukončovat neúspěšné instance strategie a rozmnožovat ty úspěšné.

Zdrojové kódy se nacházejí ve stejně pojmenovaném adresáři.

Každá strategie má své instance odlišující se parametry (instance objektu *Strategy*, soubor *Strategy.cs*). Program postupně vytvoří instance (jednu nebo více) objektu *Strategy* pro každou strategii uloženou v databázi a načte do nich potřebná data. Při tvorbě těchto instancí (v konstruktoru) je provedeno ohodnocení úspěšnosti všech jednotlivých instancí strategie (podrobněji viz. kapitola *Ohodnocování instancí* níže). Třída *Strategy* obsahuje funkci *Process*, která má na starosti ukončení horších strategií a množení těch lepších. Více v kapitolkách *Ukončování instancí* a *Křížení instancí* níže.

Výjimečně může mít strategie nastaveno, že se její proměnné vůbec nemají křížit. V takovém případě má strategie vždy právě jednu instanci. Příznak nastavující tuto vlastnost je uložen v databázi, v tabulce *T301InvestStrategy* v proměnné *geneticMixed*. Takové strategie se zpracovávají zvlášť a předchozí odstavec se jich netýká. Tyto strategie jsou postupně procházeny: Pro strategie, které ještě nemají utvořenou svoji instanci, je instance vytvořena a její proměnné jsou naplněny implicitními hodnotami, které si uživatel nastavil při nahrávání strategie do systému.

Ohodnocování instancí

Ohodnocování úspěšnosti instance provádí funkce *GetInstanceAccuracy*. Celý proces probíhá postupně pro každou strategii podle následujícího schématu:

1. Z databáze získá pro každou instanci sérii dvojic týkající se jejích odhadů:

před kolika dny se odhad konal – jaký byl rozdíl mezi skutečností a tímto odhadem

Nedostane odhady všechny, ale jen za posledních několik dní. Kolik dní, to určuje proměnná *ACCURACYMONITORPERIOD* (aktuálně má hodnotu 30), která se načítá z databáze z tabulky *T105SystemDials* (strana 65).

2. Získaná data jsou předána instanci objektu typu *InstanceAccuracyCalculator* (soubor *Strategy.cs*), který provede samotný výpočet úspěšnosti instance. Funkcí *push* se získává dvojice dat: stáří tipu (celočíslná hodnota v počtu dní, tedy v rozsahu 1..30) a jeho odchylka od reality*. *AverageAccuracyCalculator* spočítá průměrnou odchylku všech instancí za posledních několik dní (pro každý den zvlášť). Ta se dosadí místo odchylky od reality, pokud zrovna instance z nějakého důvodu svůj tip neuskutečnila. Následuje spuštění hlavní metody *getAccuracy*, která vypočte celkovou nepřesnost instance podle následujícího vzorce:

$$\sum_{\forall \text{ dvojice}(\text{stáří}, \text{tip})} \left(\frac{1}{\text{stáří}} \cdot \text{tip}^2 \right) .$$

* U měn je odchylka vždy kladná, zatímco u sportů může být i záporná (viz. níže). Platí, že čím lépe instance odhaduje, tím menší má odchylku.

Ze vzorce je zřejmé, že vyšší váhu mají chyby novějších tipů – zajímá nás aktuální přesnost tipování, nikoli jak instance tipovala dříve.

3. Pole instancí se setřídí podle jejich hodnocení – na počátku pole jsou nejméně úspěšné, na konci ty nejlepší.

Spočtení rozdílu mezi skutečností a odhadem

Sportovní strategie

Pro sportovní strategie nepočítáme rozdíly mezi kurzy, ale mezi odhadnutými pravděpodobnostmi výsledků zápasu instancí a sázkovými kancelářemi. Svou roli může sehrát také skutečný výsledek zápasu (pokud se odhad pravděpodobností výrazně liší, ale instance strategie je odhadla lépe než sázkové kanceláře).

Když chce sázková kancelář spočítat, jaké vypsát kurzy, potřebuje k tomu vědět:

- kolik chce vydělat (např. 10%, tedy $z = 0.1$)
- jaká je pravděpodobnost ohodnocovaného stavu utkání (p)

Potom se hodnota kurzu vypočítá jako: $k = \frac{1-z}{p}$.

Jak tedy z daného kurzu spočítám z a p ? Nechť jsou na zápas vypsány kurzy:

1	0	2
k_1	k_0	k_2

Pak si spočtu příslušné falešné pravděpodobnosti jako $p_{fi} = \frac{1}{k_i}$.

Pravděpodobnosti se nazývají falešné, protože jejich součet není 1 (kdyby byl, sázková kancelář by nevydělala). Tyto falešné pravděpodobnosti tedy sečtu:

$p_f = p_{f_1} + p_{f_0} + p_{f_2}$, p_f vyjde větší než 1. Z toho dokážeme spočítat z : $z = p_f - 1$. Pokud tedy vyjde například $z = 0.11$, má sázková kancelář zisk 11%.

Zbývá spočítat, jakými pravděpodobnostmi odhadli možné výsledky zápasu: $p_i = \frac{1-z}{k_i}$.

Součet těchto pravděpodobností by již měl vyjít 1. Pokud by z vyšlo menší než 0, je stanoveno na nulu. Pokud vyjde víc než 20% (0.2), je jeho hodnota stanovena jako 0.2.

Tento výpočet provedeme pro kurzy vypsány instancí strategie i pro kurzy vypsány sázkovými kancelářemi. Získáme tak odhady jednotlivých pravděpodobností obou stran (instance i sázkových kanceláří). Označme si tipy instance jako p_{ins_i} a tipy sázkových kanceláří jako p_{sk_i} . Jako výslednou odchylku instance od ideálu počítáme:

$$d = \sum_{i=1, \dots, 2} |p_{ins_i} - p_{sk_i}|,$$

ovšem může nastat výjimka: pokud instance určila vzhledem k výsledku zápasů kurzy lépe než sázkové kanceláře, rozdíl se jí nepočítá do odchylky, ale naopak se odečítá.

Měny

Pro kurzy měn je výpočet mnohem snazší, jednoduše vezmeme absolutní hodnotu rozdílu mezi skutečným kurzem a jeho odhadem.

Ukončování instancí

Ukončení špatně odhadujících instancí provádí funkce `KillWorstInstances`. Funkce prochází setříděné pole instancí od počátku (od nejhorších) a projde jich `killcount` množství. Toto číslo je vypočítáváno funkcí `GetKillCount`. Pokud není celkový počet instancí alespoň `maxInstances` neukončuje se žádná instance. Tuto hodnotu má uživatel možnost nastavit pro každou strategii a je uložena v databázi v tabulce `T301InvestStrategy`. Pokud je počet instancí strategie přesně `maxInstances`, získá se počet instancí k ukončení procentuálně z proměnné `instancesKillRatio`, nyní je tato hodnota nastavena na 25%. Pokud je instancí více, než je hodnota `maxInstances`, tak se postupuje stejně, jako v předchozím případě, ale k výslednému číslu je ještě přičten rozdíl mezi aktuálním počtem instancí a číslem `maxInstances`.

Také se nezabývá instance chráněné. Instance je chráněná po dobu prvních 5 dní (aktuální nastavení), aby vyprodukovala dostatečný počet odhadů pro zjištění, zda je instance úspěšná, či nikoli. Tuto dobu lze nastavovat pomocí proměnné `INSTANCEPROTECTPERIOD`.

Ukončení instance probíhá velmi jednoduše – v databázi je instanci přiřazeno datum jejího ukončení (pokud je datum nastaveno, je chápáno jako příznak, že instance již je ukončena) a tedy se již dále nespouští (databázová procedura `T303Kill` na straně 79).

Na konci této procedury je o každé strategii aktualizovány záznamy, kolik „platných“ instancí jí zbylo. Pokud má strategie nastavený maximální počet rodičů, kteří se mají použít při křížení instancí, je tato hodnota uložena z databáze do proměnné a příslušný počet nejlepších instancí je zkopírován do pomocného pole.

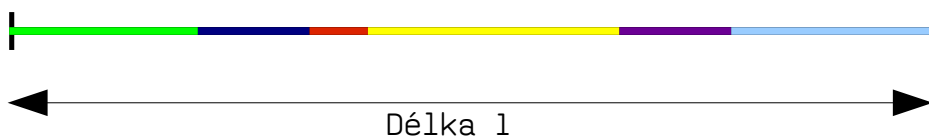
Křížení instancí

Instance množí a kříží funkce `BornNewInstance`. Proces křížení lze shrnout následovně:

1. V databázi se vytvoří nová instance
2. Nastaví se jí všechny potřebné parametry, včetně hodnot proměnných. Pokud má proměnná nastaveno, že se má křížit, o nastavení její hodnoty se stará tzv. *genetický inženýr*. Pokud má křížení zakázáno, je naplněna implicitní hodnotou.

Instancí se zrodí takový počet, aby jich strategie měla uživatelem zadaný maximální počet.

Genetický inženýr



Obrázek 15: Ilustrace normalizovaného ohodnocení instancí před výběrem vhodného rodiče – každá instance je vyznačena jinou barvou. Čím úspěšnější byla instance vyhodnocena, tím delší úsek zabírá.

Genetický inženýr má za úkol najít vhodné hodnoty proměnných nových strategií – tedy najít vhodné rodiče a zkřížit hodnoty jejich proměnných, nebo dosadit jiné hodnoty.

Hodnota proměnné může být získána třemi způsoby:

1. implicitní hodnotu zadanou uživatelem pro danou strategii
2. náhodnou hodnotu nacházející se mezi minimální a maximální možnou hodnotou (zadané uživatelem danou strategii)
3. hodnotu zděděnou po některém z rodičů

Nejprve je tedy určeno, která metoda bude použita – je voleno náhodně. S pravděpodobností `MIXINGPROBABILITY`¹ je zvoleno genetické křížení (zdědění hodnoty po jednom z možných rodičů, možnost 3). S pravděpodobností `MUTATIONPROBABILITY`² je zvolena možnost 2 a se zbylou pravděpodobností je dosazena implicitní hodnota (možnost 1).

Křížení probíhá následujícím způsobem. Nejprve najde všechny možné rodiče (všechny prosperující strategie) a jejich ohodnocení. Ohodnocení *znormalizuje* (funkce `NormalizeParents`), což znamená, že je upraví tak, aby jejich součet byl 1, ale poměr jejich ohodnocení zůstal stejný. Myšlenka úkonu je taková, že si pak vezmeme všechny znormalizované ohodnocení a naskládáme je za sebe do intervalu délky 1 (Obrázek 15). Z tohoto intervalu pak náhodně vybereme hodnotu a tím vybereme jednoho rodiče (funkce `PickParent`). Lépe ohodnocené instance v intervalu zabírají více místa, tedy logicky mají tímto způsobem vyšší šance na zvolení, což je v pořádku, protože jejich zvolení preferujeme. Interval je implementován pomocí třídy `Seedbed`.

Matematické a statistické knihovny

Do projektu jsou zahrnuté některé volně dostupné knihovny za účelem zjednodušení práce uživatelům. Není třeba, aby se uživatel při vyvíjení pluginu trápil implementací např. komplexních čísel nebo výpočtem směrodatných odchylek – je vhodnější, když se bude soustředit na samotnou myšlenku svého pluginu a pro tyto účely bez námahy využije nabídnuté knihovny.

¹ *Jméno proměnné v tabulce `T105SystemDials` (strana 65)*

² *Jméno proměnné v tabulce `T105SystemDials` (strana 65)*

Knihovny jsou uloženy v adresáři `statisticalLibraries`.

ILNumerics

ILNumerics je .NET knihovna tříd, volně dostupná, open source s *LGPL* licencí.

Implementuje velké množství numerických algoritmů všech možných druhů. Více o knihovně na webu <http://ilnumerics.net/>.

Statistická knihovna

Knihovna napsaná v C#, obsahující statistické funkce. Více informací o knihovně a její zdrojové soubory jsou k dispozici na webu <http://www.codeproject.com/KB/cs/csstatistics.aspx>.

DnAnalytics

Další numerická knihovna pro .NET. Více informací o knihovně:
<http://www.codeplex.com/dnAnalytics>.

Databáze

Ostatní komponenty systému editují obsah databáze pouze za pomoci databázových procedur, které hlídají (nebo i upravují) parametry, aby byly korektní vzhledem k integritním omezením databáze. Přímý přístup k databázi využívají ostatní pouze pro čtení – použitím dotazu `SELECT`.

Databáze slouží jako prostředek pro předávání dat mezi jednotlivými komponentami systému – kromě *BusinessDemo* aplikací ji využívají všechny ostatní komponenty.

Rejstřík databázových tabulek

T101Users.....	60	T305InvestInstanceSerializedVariables	80
T102UserAdditionInformation.....	61	T310InvestInstanceLog.....	81
T103Rights.....	63	T314InvestInstanceVariablesHistory....	83
T105SystemDials.....	65	T400InsertedMatchId.....	98
T106DataDials.....	67	T400InsertedMatchScore.....	99
T110SystemLogs.....	68	T401SuperSports.....	84
T201Money.....	69	T402Sports.....	85
T202MoneyHistory.....	71	T403Competitions.....	87
T205MoneyBets.....	72	T404CompetitionYears.....	88
T301InvestStrategy.....	74	T405Matches.....	90
T302InvestStrategyVariableRange.....	76	T406Opportunities.....	91
T303InvestInstance.....	77	T410Teams.....	93
T304InvestInstanceVariables.....	79	T420OpportunitiesBet.....	95

Uživatelé

T101Users

Tabulka uživatelů systému.

Jméno	Typ	Vlastnosti	Význam
T101Id	INT	Primární klíč	Jednoznačná identifikace
nick	VARCHAR(255)	Jedinečný, nenulový, obsahuje alespoň 4 znaky	Přezdívka uživatele používaná jako přihlašovací jméno
pwd	VARCHAR(255)	Nenulový, obsahuje alespoň 8 znaků	Heslo uživatele
T103Id	INT	Klíč do tabulky T103Rights	Práva uživatele
createAccountTime	DATETIME	Nenulový	Datum vytvoření účtu
lastLoggedTime	DATETIME	Implicitně nulové	Datum posledního přihlášení uživatele
Active	CHAR(1)	Nenulové, implicitní hodnota N, další možné hodnoty E a D	Popisuje stav aktivity uživatele. Stav může být <i>nový</i> (N) a ještě neschválený uživatel, tedy bez práv, nebo <i>povolený</i> uživatel (E), případně <i>zakázaný</i> uživatel (D). Práva se týkají možnosti uživatele spouštět na serveru své pluginy.

Kdo je používá:

Moderátor (strana 39), *Businessman* (strana 34), *Occupant* (strana 49)

Indexy:

→ Přes nick a T103Id

Přidružené funkce:

Funkce a parametry	Popis
<pre> PROCEDURE T101Insert (@nick VARCHAR(255), @pwd VARCHAR(255), @rightCode VARCHAR(32)) </pre>	Procedura přidá do databáze nového uživatele. Nick je přezdívka uživatele používaná jako login, pwd je jeho heslo a rightCode je název skupiny s právy, do které bude uživatel patřit. Pokud je rightCode null bude uživatel automaticky zařazen do anonymní skupiny. Při úspěchu vrací id nově vloženého uživatele. Návratová hodnota -1 znamená, že vkládaný a ošetřený záznam neprošel integritním omezením tabulky (nejčastější příčinou bývá krátké heslo, nebo již existující nick). Hodnota -2 znamená, že uživatel měl být zařazen do neexistující skupiny. Povinné jsou všechny parametry.
<pre> PROCEDURE T101Login (@nick VARCHAR(255), @pwd VARCHAR(255)) </pre>	Procedura přihlásí uživatele do systému. Parametry jsou stejně pojmenované jako v předchozích procedurách. Pokud je procedura přihlášení úspěšná, je aktualizován záznam lastLoggedTime a je vráceno id uživatele. Pokud nastane chyba, vrací procedura hodnotu -1. Povinné jsou oba parametry.
<pre> PROCEDURE T101Update (@T101Id INT, @newNick VARCHAR(255), @newPwd VARCHAR(255), @newRightCode VARCHAR(32), @newActive CHAR(1)) </pre>	Procedura pro aktualizaci informací o uživateli. Význam parametrů je ve stejném pořadí: id uživatele, nick uživatele, heslo uživatele, id skupiny specifikující práva uživatel a stav jeho aktivity. Povinným parametrem je pouze parametr T101Id. Ostatní parametry mohou být buď zadány a tím změněny, nebo bude na jejich místě zadána hodnota null a hodnota nebude změněna (bude uložena stávající hodnota z databáze).

T102UserAdditionInformation

Tabulka doplňujících informací o uživateli systému.

Jméno	Typ	Vlastnosti	Význam
T102Id	INT	Primární klíč	Jednoznačná identifikace
T101Id	INT	Klíč z tabulky T101Users	Položka spojující aktuální záznam s konkrétním uživatelem
firstname	VARCHAR(32)	Nenulové, délky alespoň dvou znaků	Křestní jméno uživatele
lastname	VARCHAR(32)	Nenulové, délky alespoň 2 znaky	Příjmení uživatele
sex	CHAR(1)	Nenulové, hodnoty M, nebo F	Pohlaví uživatele, male (M, muž)

nebo *female* (F, žena).

birthday	DATETIME	Implicitně nulové, od 1. 1. roku 1900 dále	Datum narození uživatele
mail	VARCHAR(128)	Nenulové, délky alespoň 6	Email uživatele
street	VARCHAR(64)	Nenulové, délky alespoň 3	Ulice, kde lze uživatele zastihnout
city	VARCHAR(64)	Nenulové, délky alespoň 2	Město, kde uživatele pobývá
psc	CHAR(5)	Nenulové, délky právě 5	Poštovní směrovací číslo adresy
territory	VARCHAR(64)	Nenulové, délky alespoň 3	Kraj (součást adresy)
Country	VARCHAR(64)	Nenulové, délky alespoň 3	Země, odkud uživatel pochází

Kdo je používá:

Moderátor (strana 39), *Businessman* (strana 34), *Occupant* (strana 49)

Indexy:

→ Přes T101Id

Přidružené funkce:

Funkce a parametry	Popis
PROCEDURE T102Insert (@T101Id INT, @firstname VARCHAR(32), @lastname VARCHAR(32), @sex CHAR(1), @birthday DATETIME, @mail VARCHAR(128), @street VARCHAR(64), @city VARCHAR(64), @psc CHAR(5), @territory VARCHAR(64), @country VARCHAR(64))	Procedura přidá do databáze nového uživatele. Jména parametrů odpovídají jménům položek v tabulce. Při úspěchu vrací hodnotu 0, pokud mají být aktualizována data týkající se uživatele, který neexistuje v tabulce T101Users, je vrácena hodnota -1, pokud jsou narušena integritní omezení, vrací hodnotu -2. Povinné jsou všechny parametry.

Funkce a parametry	Popis
<pre> PROCEDURE T102Update (@T101Id INT, @newFirstname VARCHAR(32), @newLastname VARCHAR(32), @newSex CHAR(1), @newBirthDay DATETIME, @newMail VARCHAR(128), @newStreet VARCHAR(64), @newCity VARCHAR(64), @newPsc CHAR(5), @newTerritory VARCHAR(64), @newCountry VARCHAR(64)) </pre>	Procedura pro aktualizaci informací o uživateli. Jména parametrů odpovídají jménům položek v tabulce. Při úspěchu vrací hodnotu 0, pokud mají být přidána týkající se uživatele, který neexistuje v tabulce T101Users, je vrácena hodnota -1, pokud jsou narušena integritní omezení, vrací hodnotu -2. Povinným parametrem je pouze parametr T101Id. Ostatní parametry mohou být buď zadány a tím změněny, nebo bude na jejich místě zadána hodnota null a hodnota nebude změněna (bude uložena stávající hodnota z databáze).

T103Rights

Tabulka obsahující skupiny oprávnění uživatelů.

Jméno	Typ	Vlastnosti	Význam
T103Id	INT	Primární klíč	Jednoznačná identifikace
code	VARCHAR(32)	Nenulový, délka alespoň 3, unikátní	Název skupiny
rights	VARCHAR(255)	Nenulový	Hodnota má formát deseti znaků, konkrétně pěti dvojic znaků R (čtení) a W (zápis). Dvojice mají tento význam (bráno dle jejich pořadí v řetězci): 1. čtení a zápis uživatelů a skupin prav 2. čtení a zápis všech strategií 3. čtení a zápis vlastních strategií 4. přístup k tabulce T105SystemDials 5. přístup k tabulce T106DataDials
note	VARCHAR(255)	Nenulový, implicitně prázdný řetězec	Libovolná poznámka ke skupině

Příklad

T103Id	code	rights	note
1	ADMINISTRATOR	RWRWRWRWRW	Spravce systému
2	SUPERVISOR	--R---R-RW	Prohlíží strategie
3	USER	----RW--R-	Zadává své strategie

Obrázek 16: Příklad uložení práv

Kdo je používá:

Moderátor (strana 39), Businessman (strana 34)

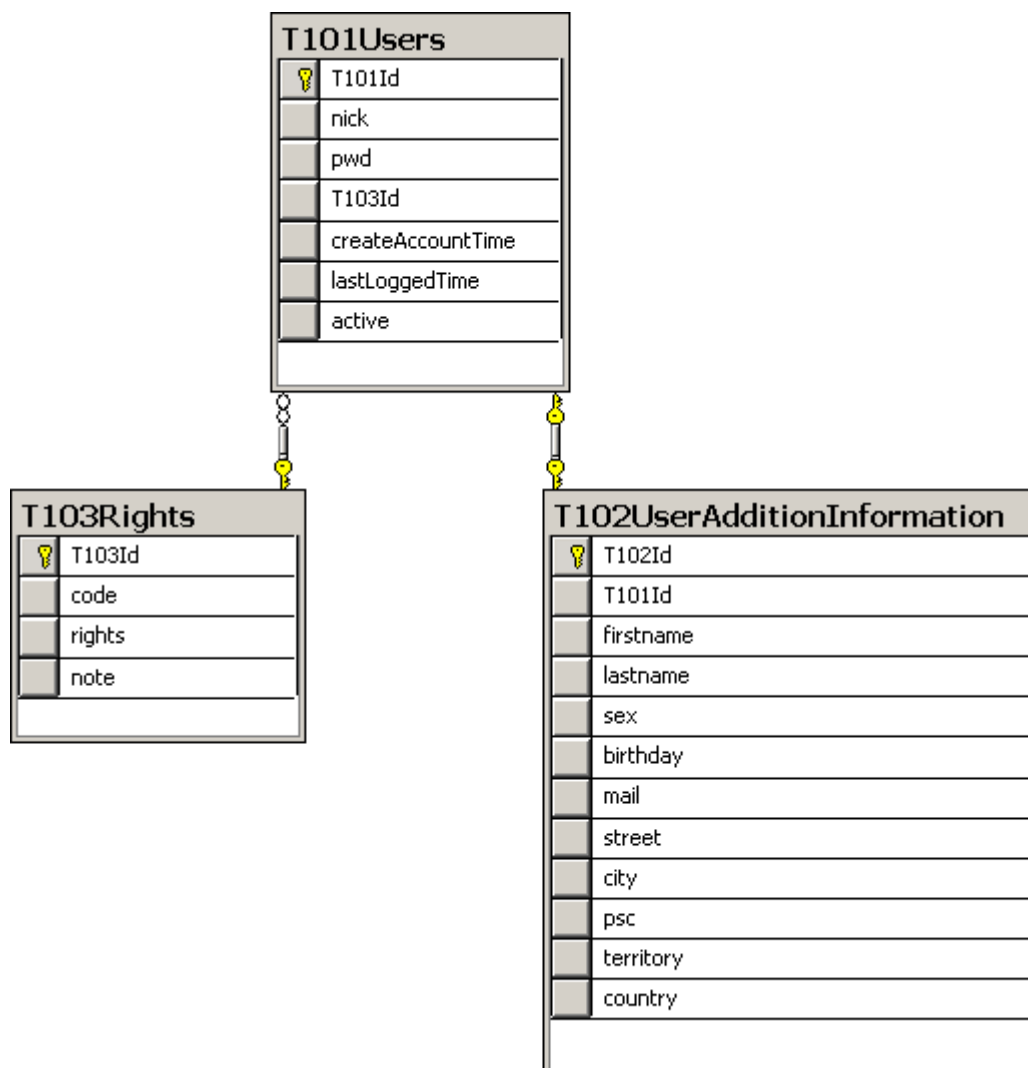
Indexy:

→ Přes code

Přidružené funkce:

Funkce a parametry	Popis
<pre>PROCEDURE T103Insert (@code VARCHAR(32), @rights VARCHAR(255), @note VARCHAR(255))</pre>	Procedura přidá do databáze novou skupinu. Jména parametrů odpovídají jménům položek v tabulce. Při úspěchu vrací kladnou hodnotu odpovídající id nově vytvořené skupiny, pokud byla porušena integritní omezení, vrací hodnotu -1. Povinné jsou všechny parametry.
<pre>PROCEDURE T103Update (@T103Id INT, @newCode VARCHAR(32), @newRights VARCHAR(255), @newNote VARCHAR(255))</pre>	Procedura pro aktualizaci informací o skupině. Jména parametrů odpovídají jménům položek v tabulce. Při úspěchu vrací hodnotu 0, pokud mají být přidána data týkající se skupiny, která neexistuje, je vrácena hodnota -1, pokud jsou narušena integritní omezení, vrací hodnotu -2. Povinným parametrem je pouze parametr T103Id. Ostatní parametry mohou být buď zadány a tím změněny, nebo bude na jejich místě zadána hodnota null a hodnota nebude změněna (bude uložena stávající hodnota z databáze).
<pre>PROCEDURE T103Delete (@T103Id INT)</pre>	Procedura pro odstranění skupiny. Procedura přijímá jediný povinný parametr - id skupiny k odstranění. Při úspěchu vrací hodnotu 0, pokud nastane chyba, je vrácena hodnota -1. Smazat lze pouze skupinu, která neobsahuje žádného uživatele.

Souhrn



Obrázek 17

Systémová data a data strategií

T105SystemDials

Tabulka pro ukládání systémových proměnných a jejich hodnot.

Jméno	Typ	Vlastnosti	Význam
T105Id	INT	Primární klíč	Jednoznačná identifikace
name	VARCHAR(32)	Nenulový, unikátní,	Název proměnné

převezeno na velká písmena

type	VARCHAR(8)	Výčtového typu: BOOLEAN, CHAR, STRING, INT, UINT, REAL	Typ proměnné
value	VARCHAR(255)		Hodnota proměnné
note	VARCHAR(255)		Libovolná poznámka

Příklad

T1...	name	type	value	note
1	MAXLOGINTIME	UINT	10	Logout time for inactive user (in minutes)
2	MAXMONEYCACHE	UINT	13	The oldest row in T203MoneyCache (in months)
3	MAXMONEYIDLE	UINT	6	Rate of money older this value dieds (in months)
4	MONEYRATEBUY	REAL	0.005	Increate/degrease price of money to buy (in percent)
5	MONEYRATESELL	REAL	-0.005	Increate/degrease price of money to sell (in percent)
6	PLUGINSPath	STRING	c:\Plugins\	Directory with all strategies
7	BUSINESSMANCOMMON	STRING	BUSINESSMANCOMMON.DLL	Inserted plugins can not have same name like BUSINESSMANCOMMON (uncase sensi...
8	THREADSCOUNT	UINT	8	Size of thread pool in Businessman
9	ACCURACY	UINT	3	Accuracy of money and sport rates (PLEASE DO NOT MODIFY, IT IS A CONSTANT!!!)
10	GENESISFULLNAME	STRING	c:\Genesis\Genesis.exe	Full name of genesis application
11	INSTANCEPROTECTPERIOD	UINT	5	Amount of days in which an invest strategy instance cannot get killed
12	MININSTANCES	UINT	10	Minimum number of live instances each strategy can have in one moment
13	MAXINSTANCES	UINT	100	Maximum number of live instances each strategy can have in one moment
14	ACCURACYMONITORPERIOD	UINT	30	Accuracy of a strategy instance is monitored up to this amount of days
16	POKUS	STRING	TRUE	nejaky pokus
18	THREADETIMEOUT	UINT	60000	Maximum timeout of each thread (in milliseconds)
19	BESTINSTANCESMIXED	UINT	10	Count of the best instances will be mixed (MAXINSTANCES <= MAXINSTANCES).
20	MONEYLIVETIMEOUT	UINT	2	Time to declare that inactive money is death (in months)

Obrázek 18: Příklad uložení systémových proměnných

Kdo je používá:

Businessman (strana 34), *Moderátor* (strana 39), *Occupant* (strana 49), *Killguard* (strana 55)

Indexy:

→ Přes name

Přidružené funkce:

Funkce a parametry	Popis
PROCEDURE T105Insert (@name VARCHAR(32), @type VARCHAR(8), @value VARCHAR(255), @note VARCHAR(255))	Procedura přidá do databáze novou proměnnou. Jména parametrů odpovídají jménům položek v tabulce. Při úspěchu vrací kladnou hodnotu odpovídající id nově vytvořené proměnné, pokud byla porušena integritní omezení, vrací hodnotu -1. Povinné jsou všechny parametry.

Funkce a parametry	Popis
<pre> PROCEDURE T105Update (@T105Id INT, @newName VARCHAR(32), @newType VARCHAR(8), @newValue VARCHAR(255), @newNote VARCHAR(255)) </pre>	Procedura pro aktualizaci dat o proměnné. Jména parametrů odpovídají jménům položek v tabulce. Při úspěchu vrací hodnotu 0, pokud mají být upravena data proměnné, která neexistuje, je vrácena hodnota -1, pokud jsou narušena integritní omezení, vrací hodnotu -2. Povinným parametrem je pouze parametr T105Id. Ostatní parametry mohou být buď zadány a tím změněny, nebo bude na jejich místě zadána hodnota null a hodnota nebude změněna (bude uložena stávající hodnota z databáze).
<pre> PROCEDURE T105Delete (@T105Id INT) </pre>	Procedura pro odstranění proměnné z databáze. Parametrem je id záznamu. Při úspěchu vrací hodnotu 0, pokud se mazání nezdařilo je vrácena hodnota -1.
<pre> TRIGGER T105TrgForInsertUpdate ON T105SystemDials FOR INSERT, UPDATE </pre>	Kontroluje hodnoty proměnných dle jejich typu.

T106DataDials

Tabulka pro ukládání datových parametrů pluginů, uživatelů atd. Tabulka je momentálně nevyužívaná, určená pro budoucí použití.

Jméno	Typ	Vlastnosti	Význam
T105Id	INT	Primární klíč	Jednoznačná identifikace
name	VARCHAR(32)	Nenulový, unikátní, převedeno na velká písmena	Název proměnné
type	VARCHAR(8)	Výčtového typu: BOOLEAN, CHAR, STRING, INT, UINT, REAL	Typ proměnné
value	VARCHAR(255)		Hodnota proměnné
note	VARCHAR(255)		Libovolná poznámka

Kdo je používá:

Momentálně ji nevyužívá žádná část projektu.

Indexy:

➔ Přes name

Přidružené funkce:

Funkce a parametry	Popis
<pre> PROCEDURE T106Insert (@name VARCHAR(32), @type VARCHAR(8), @value VARCHAR(255), @note VARCHAR(255)) </pre>	Procedura přidá do databáze novou proměnnou. Jména parametrů odpovídají jménům položek v tabulce. Při úspěchu vrací kladnou hodnotu odpovídající id nově vytvořené proměnné, pokud byla porušena integritní omezení, vrací hodnotu -1. Povinné jsou všechny parametry.
<pre> PROCEDURE T106Update (@T106Id INT, @newName VARCHAR(32), @newType VARCHAR(8), @newValue VARCHAR(255), @newNote VARCHAR(255)) </pre>	Procedura pro aktualizaci dat o proměnné. Jména parametrů odpovídají jménům položek v tabulce. Při úspěchu vrací hodnotu 0, pokud mají být upravena data proměnné, která neexistuje, je vrácena hodnota -1, pokud jsou narušena integritní omezení, vrací hodnotu -2. Povinným parametrem je pouze parametr T106Id. Ostatní parametry mohou být buď zadány a tím změněny, nebo bude na jejich místě zadána hodnota null a hodnota nebude změněna (bude uložena stávající hodnota z databáze).
<pre> PROCEDURE T106Delete (@T106Id INT) </pre>	Procedura pro odstranění proměnné z databáze. Parametrem je id záznamu. Při úspěchu vrací hodnotu 0, pokud se mazání nezdařilo je vrácena hodnota -1.
<pre> TRIGGER T106TrgForInsertUpdate ON T106SystemDials FOR INSERT, UPDATE </pre>	Kontroluje hodnoty proměnných dle jejich typu.

T110SystemLogs

Tabulka pro ukládání systémových logů.

Jméno	Typ	Vlastnosti	Význam
T110Id	INT	Primární klíč	Jednoznačná identifikace
head1	VARCHAR(32)	Nenulový, délka alespoň 3 znaky, převedeno na velká písmena	Hlavní název logů
head2	VARCHAR(32)	Nenulový, délka alespoň 3 znaky, převedeno na velká písmena	Rozšířený název logů
time	DATETIME	Nenulový, od 1.1.1990 dále	Datum a čas uložení logů, tedy téměř přesně čas vzniku hlášení
message	VARCHAR(255)	Nenulové délky	Samotný text hlášení

Kdo je používá:

Businessman (strana 34)

Indexy:

→ Přes head1, head2 a time

Přidružené funkce:

Funkce a parametry	Popis
<pre>PROCEDURE T110Insert (@head1 VARCHAR(32), @head2 VARCHAR(32), @message VARCHAR(255))</pre>	Procedura přidá do databáze nové hlášení. Jména parametrů odpovídají jménům položek v tabulce. Při úspěchu vrátí kladnou hodnotu odpovídající id přidaného řádku. Pokud se uložení nepovedlo, vrátí hodnotu -1. Povinné jsou všechny parametry.

Souhrn

T110SystemLogs	
	T110Id
	head1
	head2
	time
	message

T106DataDials	
	T106Id
	name
	type
	value
	note

T105SystemDials	
	T105Id
	name
	type
	value
	note

Obrázek 19

Kurzy měn

T201Money

Tabulka měn.

Jméno	Typ	Vlastnosti	Význam
T201Id	SMALLINT	Primární klíč	Jednoznačná identifikace
code	CHAR(3)	Jedinečný, nenulový, velkými písmeny	Kód měny (např. EUR, USD, JPY)
bornTime	DATETIME	Nenulový	Od kdy je měna evidována

Příklad:

T201Id	code	bornTime	deathTime
1	CZK	1/1/1991 12:00:00 AM	NULL
2	ATS	1/1/1991 12:00:00 AM	12/31/2001 12:00:00 AM
3	AUD	1/1/1991 12:00:00 AM	NULL
4	BEC	1/1/1991 12:00:00 AM	1/24/1991 12:00:00 AM
5	CAD	1/1/1991 12:00:00 AM	NULL
6	CHF	1/1/1991 12:00:00 AM	NULL

Obrázek 20: Příklad uložení informací o měnách

Kdo je používá:

Businessman (strana 34), *Occupant* (strana 49), *MoneyDownload* (strana 12)

Indexy:

→ Přes code

Přidružené funkce:

Funkce a parametry	Popis
<pre>PROCEDURE T201Insert (@code CHAR(3), @bornTime DATETIME)</pre>	Procedura přidá do databáze novou měnu. Jména parametrů odpovídají jménům položek v tabulce. Při úspěchu vrací kladnou hodnotu odpovídající id přidané měny. Pokud se uložení nepovedlo (nevyhovuje integritním omezením), vrací hodnotu -1. Povinný je pouze první parametr. Pokud je hodnota druhého parametru null, do databáze je uložen aktuální čas.
<pre>FUNCTION T201GetId (@code CHAR(3)) RETURNS SMALLINT</pre>	Funkce pro získání primárního klíče (T201Id) dané měny. Funkce vrací číselnou hodnotu primárního klíče, pokud takový neexistuje, vrací hodnotu null. Parametr code je povinný.
<pre>PROCEDURE T201Death (@T201Id INT, @deathTime DATETIME)</pre>	Procedura pro nastavení (předpokládaného) data zániknutí měny. Při úspěchu vrací hodnotu 0, při chybě (záznam nenalezen) hodnotu -1. Povinné jsou oba parametry. Pokud druhý parametr obsahuje datum, bude uloženo. Pokud obsahuje hodnotu null, bude uloženo datum zrušení měny odstraněno.

T202MoneyHistory

Tabulka kurzů měn.

Jméno	Typ	Vlastnosti	Význam
T202Id	INT	Primární klíč	Jednoznačná identifikace
T201IdFrom	SMALLINT	Klíč T201Id z tabulky T201Money	Kód měny pro převod (ze které měny převádíme)
T201IdTo	SMALLINT	Klíč T201Id z tabulky T201Money	Kód měny pro převod (do které měny převádíme)
time	DATETIME	Od 1.1.1990 dále	Kdy byl tento kurz
value	DECIMAL(24, 8)	Větší než nula	Hodnota kurzu

Hodnoty T201IdFrom, T201IdTo a time tvoří unikátní trojici tabulky.

Příklad:

T202Id	T201IdFrom	T201IdTo	time	value
1	1	2	1/1/1991 12:00:00 AM	2.60000000
2	2	1	1/1/1991 12:00:00 AM	0.38461538
3	1	3	1/1/1991 12:00:00 AM	21.39000000
4	3	1	1/1/1991 12:00:00 AM	0.04675082
5	2	3	1/1/1991 12:00:00 AM	8.22692308

Obrázek 21: Příklad uložení dat o kurzech měn

Z obrázku 20 na straně 70 a obrázku 21 vidíme, že kurz CZK (id 1) vůči ATS (id 2) byl 1.1.1991 hodnoty 2,6.

Kdo je používá:

MoneyDownload (strana 12), *Businessman* (strana 34), *Occupant* (strana 49), *Killguard* (strana 55)

Indexy:

➔ Přes T201IdFrom a T201IdTo

Přidružené funkce:

Funkce a parametry	Popis
PROCEDURE T202Insert (@codeFrom CHAR(3), @codeTo CHAR(3), @time DATETIME, @value DECIMAL(24, 8))	Procedura přidá kurz mezi měnami. Pokud měny neexistují, jsou vytvořeny do tabulky T201Money. Jména parametrů odpovídají jménům položek v tabulce. Vrací 0, pokud se operace vložení zdařila, jinak záporné číslo chyby. Pokud je vrácena hodnota -1, jedná se o porušení integritních omezení. Povinné jsou všechny parametry.

T205MoneyBets

Tabulka odhadů kurzů mezi měnami jednotlivými instancemi strategií. Pokud bude zrušena instance vlastníci tyto odhady, bude tabulka zrušena.

Jméno	Typ	Vlastnosti	Význam
T205Id	INT	Primární klíč	Jednoznačná identifikace
T303Id	INT	Klíč T303Id z tabulky T303InvestInstance, nenulový	Spojení mezi záznamem a konkrétní instancí
T201IdFrom	SMALLINT	Klíč T201Id z tabulky T201Money	Z které měny
T201IdTo	SMALLINT	Klíč T201Id z tabulky T201Money	Na kterou měnu
time	DATETIME	Od 1.1.1990 dále	Čas uložení tipu kurzu
bet	DECIMAL(24, 8)	Větší, než nula	Vlastní hodnota odhadu kurzu

Čtveřice T303Id, T201IdFrom, T201IdTo a time je unikátní v rámci tabulky.

Příklad:

	T205Id	T303Id	T201IdFrom	T201IdTo	time	bet
1		1	1	22	1/6/2009 3:00:01 AM	17.93359947
2		2	1	22	1/6/2009 3:00:01 AM	17.93359947
3		3	1	22	1/6/2009 3:00:01 AM	17.93359947
4		4	1	22	1/6/2009 3:00:01 AM	17.93359947
5		5	1	22	1/6/2009 3:00:01 AM	17.93359947

Obrázek 22: Příklad uložení sázek

Instance s id 1 odhadovala 6.1.2009 kurz mezi CZK (id 1) a USD (id 22) na 7.1.2009 hodnotou 17,93359947.

Kdo je používá:

MoneyDownload (strana 12), *Businessman* (strana 34), *Occupant* (strana 49), *Killguard* (strana 55)

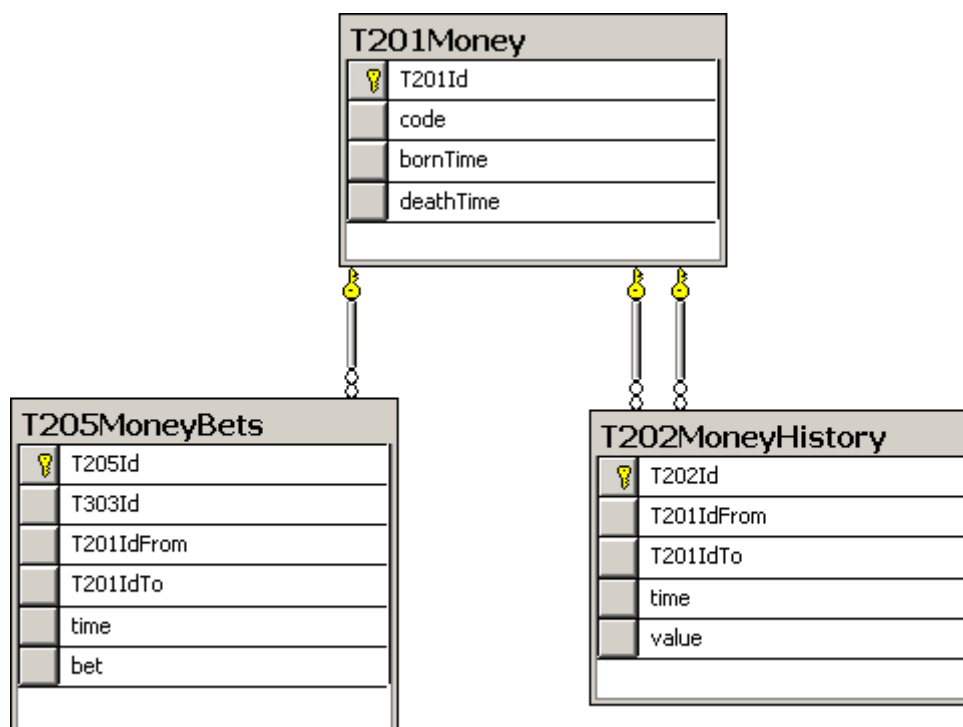
Indexy:

→ Přes T301Id a (T303Id, T201IdFrom, T201IdTo, time)

Přidružené funkce:

Funkce a parametry	Popis
<pre>PROCEDURE T205InsertOrUpdate (@T303Id INT, @T201IdFrom INT, @T201IdTo INT, @time DATETIME, @bet DECIMAL(24, 8))</pre>	Procedura uloží odhad kurzu mezi měnami. Jména parametrů odpovídají jménům položek v tabulce. Vrací id řádku, který byl přidán, nebo změněn. Pokud se procedura nezdaří a jednalo se o vložení, je vrácena hodnota -1, pokud se jednalo o neúspěšný pokus modifikace, vrací hodnotu -2. Povinné jsou všechny parametry.

Souhrn



Obrázek 23

Data o strategiích a jejich instancích

T301InvestStrategy

Tabulka jednotlivých strategií.

Jméno	Typ	Vlastnosti	Význam
T301Id	INT	Primární klíč	Jednoznačná identifikace
T101Id	INT	Klíč T101Id z tabulky T101Users, nenulový	Spojení mezi strategií a uživatelem, který ji do systému nahrál
title	VARCHAR(64)	Délky alespoň 4 znaky, unikátní	Název strategie, který si volí uživatel
insertTime	DATETIME	Od 1.1.1990 dále	Datum a čas vytvoření strategie v systému (nikoli aktualizace, ale vytvoření)
kind	VARCHAR(8)	Výčtový typ: MONEY, SPORT	Jaké odhady strategie počítá – sportovní, nebo měnové
language	VARCHAR(16)	Výčtový typ: C#, VISUAL BASIC, DELPHI	Jazyk, ve kterém je plugin (DLL knihovna) implementovaný
state	VARCHAR(12)	Výčtový typ: CONSTRUCTING, LIVE, ZOMBIE Implicitní hodnota je CONSTRUCTING	Stav, ve kterém se plugin nachází – ve vývoji, standardně běžící, nebo již nepoužívaný
geneticMixed	CHAR(1)	Výčtový typ: Y, N	Hodnota udávající, zda se v pluginu budou křížit hodnoty proměnných. Y znamená ano, N znamená ne.
maxInstances	INT	Nenulový, v uzavřeném intervalu 1, 100	Maximální počet instancí dané strategie
mixedInstances	INT	Nenulový, v uzavřeném intervalu 1, 100	Počet nejlepších instancí, které se budou křížit (pokud je křížení povoleno)

Dvojice T101Id a title jsou v rámci tabulky unikátní.

Příklad:

T301Id	T101Id	title	insertTime	kind	lan...	state	note
1	4	testovací_průměr	12/8/2008 12:34:24 PM	MONEY	C#	LIVE	
2	4	testovací_sport	12/8/2008 12:35:08 PM	SPORT	C#	LIVE	
8	3	Random s odchylkou B	12/13/2008 10:59:05 PM	MONEY	C#	LIVE	1. Random
9	4	testovani logovani na penezich	1/30/2009 9:20:20 PM	MONEY	C#	ZOMBIE	
10	4	log2	1/30/2009 9:21:58 PM	MONEY	C#	LIVE	
11	4	test	1/30/2009 9:52:59 PM	MONEY	C#	CONSTRUCTING	

Obrázek 24: Příklad uložení strategií

Po vložení strategie je automaticky nastaven state na CONSTRUCTING. Poté se ve složce c:\MoneyMaker\InvestStrategy\ vytvoří soubor se jménem id.dll, kde id je *Id* vkládané strategie. Dále se nastaví parametry u strategie a potom lze strategii spustit (nastavit state na LIVE). Strategie se nedá smazat, místo toho se jejich stav nastavuje na hodnotu ZOMBIE.

Kdo je používá:

Occupant (strana 49), *Businessman* (strana 34), *Killguard* (strana 55)

Indexy:

➔ Přes T101Id

Přidružené funkce:

Funkce a parametry	Popis
<pre>PROCEDURE T301Insert (@T101Id INT, @title VARCHAR(64), @kind VARCHAR(8), @language VARCHAR(16), @note VARCHAR(255), @geneticMixed CHAR(1), @maxInstances INT, @mixedInstances INT)</pre>	Procedura uloží novou strategii. Jména parametrů odpovídají jménům položek v tabulce. Vrací id přidané strategie. Pokud se procedura nezdaří, je vrácena hodnota -1 (porušení integritních omezení). Povinné jsou parametry T101Id, title, kind, language a note. Hodnoty zbylých proměnných mohou být nahrazeny hodnotou null a budou vyplněny implicitními hodnotami.
<pre>CREATE PROCEDURE T301Update (@T301Id INT, @newT101Id INT, @newTitle VARCHAR(64), @newKind VARCHAR(8), @newLanguage VARCHAR(16), @newState VARCHAR(12), @newNote VARCHAR(255), @newGeneticMixed CHAR(1), @newMaxInstances INT, @newMixedInstances INT)</pre>	Procedura upraví již vytvořenou strategii. Jména parametrů odpovídají jménům položek v tabulce. Vrací 0 při úspěchu. Pokud se procedura nezdaří, protože strategie nebyla podle svého id nalezena, je vrácena hodnota -1. Při porušení integritních omezení je navracena hodnota -2. Povinný je pouze parametr T301Id. Zbylé parametry mohou být nahrazeny hodnotou null, aby jejich hodnota zůstala nezměněna.

T302InvestStrategyVariableRange

Tabulka rozsahu hodnot jednotlivých proměnných strategií. Pokud bude odpovídající investiční strategie zrušena, budou zrušeny i její proměnné.

Jméno	Typ	Vlastnosti	Význam
T302Id	INT	Primární klíč	Jednoznačná identifikace
T301Id	INT	Klíč T301Id z tabulky T301InvestStrategy, nenulový	Spojení mezi proměnnými a strategií.
name	VARCHAR(32)	Délky alespoň 1 znak, unikátní, vždy překonvertován na velká písmena	Název proměnné
type	VARCHAR(16)	Výčtový typ: BOOLEAN, CHAR, INTEGER, UNSIGNED INTEGER, FLOAT, DOUBLE, DECIMAL, STRING, LONG	Typ proměnné
mixed	CHAR(1)	Výčtový typ: E, D Implicitní hodnota je E	Může se proměnná křížit?
minValue	VARCHAR(255)	Nenulová	Minimální hodnota proměnné (při křížení) včetně
defaultValue	VARCHAR(255)	Nenulová	Preferovaná (iniciální) hodnota proměnné (při křížení)
maxValue	VARCHAR(255)	Nenulová	Maximální hodnota proměnné (při křížení) včetně
note	VARCHAR(255)	Nenulový, implicitně prázdný řetězec	Poznámka k proměnné

Dvojice T301Id a name jsou v rámci tabulky unikátní.

Příklad:

T302Id	T301Id	name	type	mixed	minValue	defaultValue	maxValue	note
1	1	D2	DOUBLE	E	45	45	45	
2	1	DAYS	UNSIGNED INTEGER	E	1	1000	5	

Obrázek 25: Příklad uložení proměnných a jejich vlastností

Strategie s id 1 má proměnnou nazvanou D2, která je typu DOUBLE a je povoleno ji křížit. Její minimální hodnota je 45, stejně tak jako preferovaná a maximální hodnota. (Ve výsledku se tedy křížit nebude, protože její hodnotu nelze měnit.)

Kdo je používá:

Occupant (strana 49), *Businessman* (strana 34), *Killguard* (strana 55)

Indexy:

→ Přes T301Id a name

Přidružené funkce:

Funkce a parametry	Popis
<pre>PROCEDURE T302Insert (@T301Id INT, @name VARCHAR(32), @type VARCHAR(16), @mixed CHAR(1), @minValue VARCHAR(255), @defaultValue VARCHAR(255), @maxValue VARCHAR(255), @note VARCHAR(255))</pre>	Procedura uloží novou proměnnou strategie. Jména parametrů odpovídají jménům položek v tabulce. Vrací id přidané proměnné. Pokud se procedura nezdaří, je vrácena hodnota -1 (porušení integritních omezení). Povinné jsou všechny parametry.
<pre>PROCEDURE T302Update (@T302Id INT, @newT301Id INT, @newName VARCHAR(32), @newType VARCHAR(16), @newMixed CHAR(1), @newMinValue VARCHAR(255), @newDefaultValue VARCHAR(255), @newMaxValue VARCHAR(255), @newNote VARCHAR(255))</pre>	Procedura upraví již vytvořenou proměnnou. Jména parametrů odpovídají jménům položek v tabulce. Vrací 0 při úspěchu. Pokud se procedura nezdaří, protože proměnná nebyla podle svého id nalezena, je vrácena hodnota -1. Při porušení integritních omezení je navracena hodnota -2. Povinný je pouze parametr T302Id. Zbylé parametry mohou být nahrazeny hodnotou null, aby jejich hodnota zůstala nezměněna.
<pre>PROCEDURE T302Delete (@T302Id INT)</pre>	Procedura pro odstranění proměnné z databáze. Povinným parametrem je její id. Vrací hodnotu 0 při úspěchu, -1 při chybě.

T303InvestInstance

Tabulka jednotlivých instancí dané strategie.

Jméno	Typ	Vlastnosti	Význam
T303Id	INT	Primární klíč	Jednoznačná identifikace
T301Id	INT	Klíč T301Id z tabulky T301InvestStrategy, nenulový	Spojení mezi instancí a strategií.
bornTime	DATETIME	Nenulové, od 1.1.1990 dále	Čas vzniku instance
deathTime	DATETIME	Implicitně nulové	Čas zániku instance (pokud je

null, instance stále žije)

accuracy	REAL	Implicitně 0, nikdy NULL	Nepřesnost v odhadování instance (negativní hodnocení její přesnosti)
snub	INT	Implicitně 0, nikdy NULL	Počet „napomenutí“ instance. Napomenutí se udělují, pokud instance vyvolává chybové výjimky. Uděleno je maximálně jedno napomenutí denně. Pokud je počet napomenutí vyšší než udává systémová proměnná SNUBSBEFOREKILL, je instance ukončena.

Příklad:

T303Id	T301Id	bornTime	deathTime	accuracy	snub
355	4	14.5.2009 7:00:07	NULL	0	1
356	4	14.5.2009 7:00:07	NULL	0	1
357	4	14.5.2009 7:00:07	NULL	0	1

Obrázek 26: Příklad uložení dat o instancích – zde vidíme instance, které zatím jednou spadly

Na obrázku 26 vidíme instance strategie s id 1. Instance s id 62 a 63 ještě žijí, instance s id 64 a 65 již byly zrušeny.

Kdo je používá:

Occupant (strana 49), *Businessman* (strana 34), *Killguard* (strana 55)

Indexy:

→ Přes T301Id

Přidružené funkce:

Funkce a parametry	Popis
PROCEDURE T303Born (@T301Id INT)	Procedura, která vytváří instanci strategie. Pozor, negeneruje proměnné. Jako povinný a jediný parametr přijímá id strategie. Vrací id vytvořené instance. Pokud se procedura nezdaří, je vrácena hodnota -1 (porušení integritních omezení).

Funkce a parametry	Popis
<pre>PROCEDURE T303Kill (@T303Id INT)</pre>	Procedura, která ruší instanci strategie. Jako povinný a jediný parametr přijímá id dotčené instance. Vrací hodnotu 0 při úspěchu. Pokud se procedura nezdaří, je vrácena hodnota -1 (nenalezená instance).
<pre>PROCEDURE T303SetAccuracy (@T303Id INT, @accuracy REAL)</pre>	Procedura pro nastavení nové hodnoty úspěšnosti. Oba parametry jsou povinné (číslo instance, nová úspěšnost). Vrací hodnotu 0 při úspěchu. Pokud se procedura nezdaří, je vrácena hodnota -1 (nenalezená instance).
<pre>PROCEDURE T303SetSnub (@T303Id INT, @snub INT)</pre>	Procedura pro nastavení nebo zvýšení počtu napomenutí instance. Oba parametry jsou povinné (číslo instance, počet napomenutí). Pokud je druhý parametr nastaven na 0, je počítadlo vynulováno, jinak je zvýšeno o jedna. Vrací hodnotu 0 při úspěchu. Pokud se procedura nezdaří, je vrácena hodnota -2 (nenalezená instance), nebo -1 (pokud se nepodařilo instanci ukončit).

T304InvestInstanceVariables

Tabulka křížitelných proměnných a jejich aktuálních hodnot pro instance strategií.

Jméno	Typ	Vlastnosti	Význam
T304Id	INT	Primární klíč	Jednoznačná identifikace
T303Id	INT	Klíč T303Id z tabulky T303InvestInstance, nenulový	Spojení mezi proměnnou a instancí.
name	VARCHAR(32)	Nenulové, délka alespoň 1 znak	Jméno proměnné
value	VARCHAR(255)		Hodnota proměnné

Dvojice proměnných T303Id a name je unikátní v rámci tabulky.

Příklad:

	T304Id	T303Id	name	value
1		1	D2	45
2		1	DAYS	1000
3		2	D2	45
4		2	DAYS	1000

Obrázek 27: Příklad uložení hodnot proměnných instancí

Kdo je používá:

Businessman (strana 34)

Indexy:

→ Přes T303Id a name

Přidružené funkce:

Funkce a parametry	Popis
<pre>PROCEDURE T304InsertOrUpdate (@T303Id INT, @name VARCHAR(32), @value VARCHAR(255))</pre>	Procedura, která vytvoří, nebo aktualizuje hodnotu proměnné. Jména parametrů odpovídají jménům uvedeným v předchozí tabulce. Vrací id přidané, nebo upravené, proměnné. Pokud se nezdaří vytváření nové proměnné, je vrácena hodnota -1. Pokud se nepovedla aktualizovat stávající hodnota, je vrácena hodnota -2. Povinné parametry jsou T303Id a name. Pokud je proměnná value nastavena na hodnotu null, nebude hodnota v databázi změněna.
<pre>CREATE TRIGGER T304TrgAfterInsertUpdate ON T304InvestInstanceVariables AFTER INSERT, UPDATE</pre>	Trigger pro logování hodnot proměnných v čase – pokud se změní hodnota proměnné, bude tato událost zalogována. Logy se ukládají do tabulky T314InvestInstanceVariablesHistory (strana 83).

T305InvestInstanceSerializedVariables

Tabulka proměnných, které se nekříží, a jejich aktuálních hodnot pro instance strategií.

Jméno	Typ	Vlastnosti	Význam
T305Id	INT	Primární klíč	Jednoznačná identifikace
T303Id	INT	Klíč T303Id z tabulky T303InvestInstance, nenulový	Spojení mezi proměnnou a instancí.
name	VARCHAR(32)	Nenulové, délky alespoň 1	Jméno proměnné

value

VARCHAR(255)

Hodnota proměnné

Dvojice proměnných T303Id a name je unikátní v rámci tabulky.

Příklad:

T305Id	T303Id	name	value
1	1	P1	<Binary data>
2	1	P2	<Binary data>
3	1	P3	<Binary data>
4	1	P4	<Binary data>
5	1	doubles	<Binary data>
6	2	P1	<Binary data>

Obrázek 28: Příklad uložení hodnot proměnných instancí

Kdo je používá:

Businessman (strana 34)

Indexy:

→ Přes T303Id a name

Přidružené funkce:

Funkce a parametry	Popis
PROCEDURE T305InsertOrUpdate (@T303Id INT , @name VARCHAR (32), @value VARBINARY (MAX))	Procedura, která vytvoří, nebo aktualizuje hodnotu proměnné. Jména parametrů odpovídají jménům uvedeným v předchozí tabulce. Vrací id přidané, nebo upravené, proměnné. Pokud se nezdaří vytváření nové proměnné, je vrácena hodnota -1. Pokud se nepovedla aktualizovat stávající hodnota, je vrácena hodnota -2. Povinné parametry jsou T303Id a name. Pokud je proměnná value nastavena na hodnotu null, nebude hodnota v databázi změněna.

T310InvestInstanceLog

Tabulka obsahující logy instancí.

Jméno	Typ	Vlastnosti	Význam
T310Id	INT	Primární klíč	Jednoznačná identifikace

T303Id	INT	Klíč T303Id z tabulky T303InvestInstance, nenulový	Spojení mezi logem a instancí.
time	DATETIME	Nenulový, od 1.1.1990 dále	Čas, kdy byl log zapsán
message	VARCHAR(255)		Zpráva od instance

Příklad:

T310Id	T303Id	time	message
396	306	22.4.2009 3:00:08	System.IndexOutOfRangeException: Index was outside the bounds of the array. □□ at BusinessmanPlugin.Plugin.run(PackBase pack)□□ at BusinessmanC...
397	426	22.4.2009 3:00:08	DOWNLARGE won
398	307	22.4.2009 3:00:08	Jan Sport commencing...
399	510	22.4.2009 3:00:08	System.ArgumentOutOfRangeException: Index was out of range. Must be non-negative and less than the size of the collection. □□Parameter name: index□...
400	426	22.4.2009 3:00:08	Predicting change of currency by -0.00677918
401	73	22.4.2009 3:00:08	run function started
402	307	22.4.2009 3:00:08	Lowest ranked team: 132, Highest ranked team: 140
403	307	22.4.2009 3:00:08	Future matches count: 16
404	511	22.4.2009 3:00:08	System.ArgumentOutOfRangeException: Index was out of range. Must be non-negative and less than the size of the collection. □□Parameter name: index□...
405	307	22.4.2009 3:00:08	System.IndexOutOfRangeException: Index was outside the bounds of the array. □□ at BusinessmanPlugin.Plugin.run(PackBase pack)□□ at BusinessmanC...
406	74	22.4.2009 3:00:08	run function started
407	427	22.4.2009 3:00:08	Jan Money Strategy commencing
408	308	22.4.2009 3:00:08	Jan Sport commencing...
409	512	22.4.2009 3:00:08	System.ArgumentOutOfRangeException: Index was out of range. Must be non-negative and less than the size of the collection. □□Parameter name: index□...
410	308	22.4.2009 3:00:08	Lowest ranked team: 132, Highest ranked team: 140

Obrázek 29: Příklad uložení logů instancí

Kdo je používá:

Businessman (strana 34), *Occupant* (strana 49)

Indexy:

→ Přes T303Id

Přidružené funkce:

Funkce a parametry	Popis
PROCEDURE T310Insert (@T303Id INT , @message VARCHAR(255))	Procedura, která přidá do logu hlášení. Jména parametrů odpovídají jménům uvedeným v předchozí tabulce. Vrací id přidávaného logu. Pokud se operace nezdaří, je vrácena hodnota -1. Povinné parametry jsou oba parametry.

T314InvestInstanceVariablesHistory

Tabulka obsahující historické hodnoty proměnných instancí.

Jméno	Typ	Vlastnosti	Význam
T314Id	INT	Primární klíč	Jednoznačná identifikace
T304Id	INT	Klíč T304Id z tabulky T304InvestInstanceVariables, nenulové	Spojení mezi hodnotou a proměnnou.
time	DATETIME	Nenulový, od 1.1.1990 dále	Čas, kdy proměnná uloženou hodnotu získala
value	VARCHAR(255)	Nenulová	Získaná hodnota proměnné

Příklad:

T314Id	T304Id	time	value
1	1	12/15/2008 3:00:01 AM	45
2	201	12/15/2008 3:00:01 AM	5
3	202	12/15/2008 3:00:01 AM	'dfdsd'sdf'
4	203	12/15/2008 3:00:01 AM	12

Obrázek 30: Příklad uložení starých hodnot proměnných

Kdo je používá:

Businessman (strana 34)

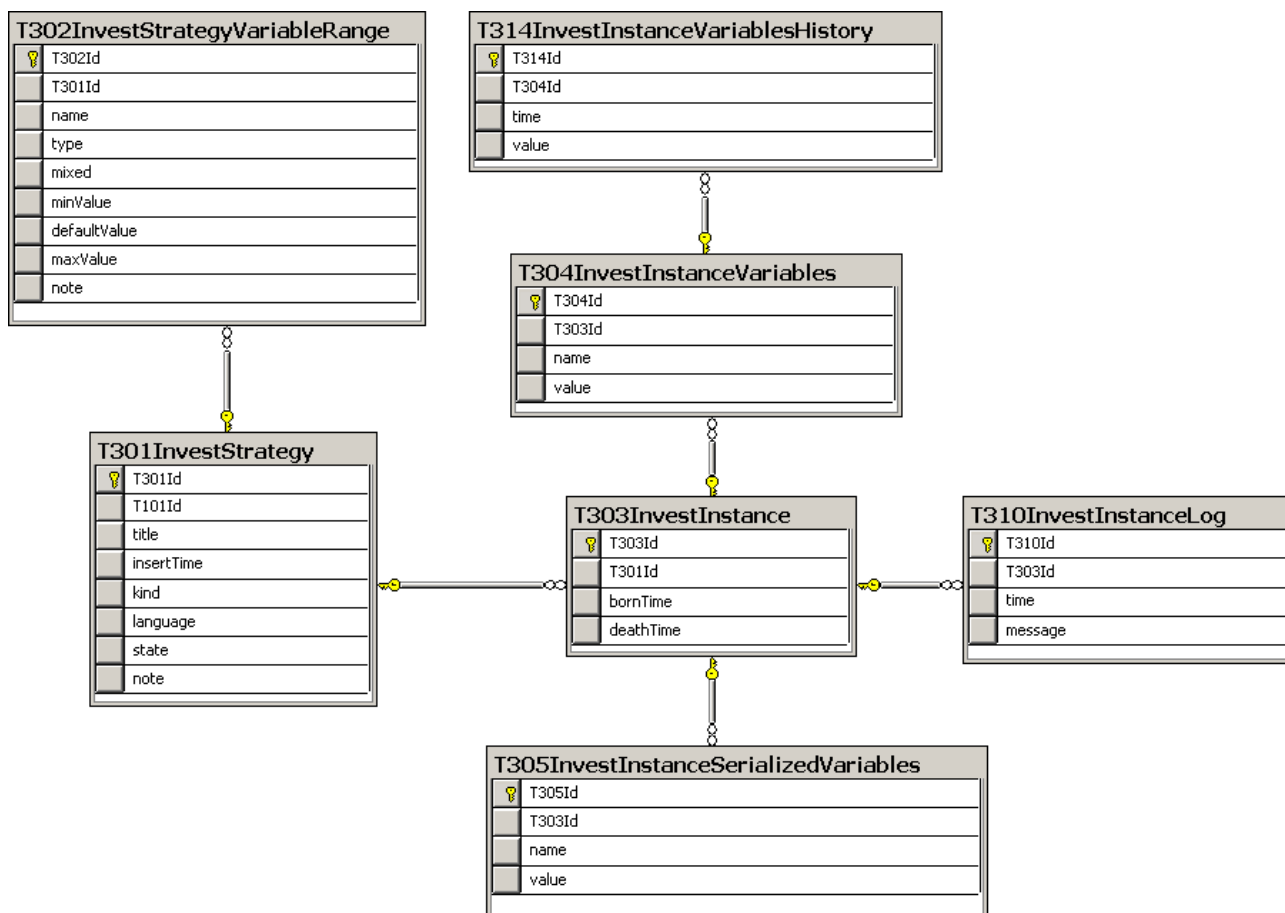
Indexy:

➔ Přes T304Id

Přidružené funkce:

žádné

Souhrn



Obrázek 31

Sportovní data

Než se pustíte do čtení, je vhodné si přečíst krátký úvod osvětlující použitou terminologii, abyste snadněji porozuměli jménům proměnných v tabulkách. Terminologie je popsána v kapitole *Terminologie* na straně 23.

T401SuperSports

Tabulka hlavních druhů sportů.

Jméno	Typ	Vlastnosti	Význam
T401Id	INT	Primární klíč	Jednoznačná identifikace
Title	VARCHAR(64)	Jedinečný, nenulový, velkými písmeny	Jméno sportu, např. HOKEJ, BASKETBAL...

Příklad:

T401Id	title
13	BASKETBAL
12	FOTBAL
14	HAZENA
11	HOKEJ

Obrázek 32: Příklad uložení "super sportu"

Kdo je používá:

Sports Cribber (strana 15), *Businessman* (strana 34)

Indexy:

→ Přes title

Přidružené funkce:

Funkce a parametry	Popis
<pre>PROCEDURE T401Insert (@title VARCHAR(255))</pre>	Procedura, která přidá nový „super sport“. Jako jediný parametr vyžaduje jméno nového sportu. Vrací id přidaného záznamu. Pokud se operace nezdaří, je vrácena hodnota -1.
<pre>PROCEDURE T401Update (@T401Id INT, @newTitle VARCHAR(255))</pre>	Procedura upraví již vytvořený „super sport“. Jména parametrů odpovídají jménům položek v tabulce. Vrací 0 při úspěchu. Pokud se procedura nezdaří, protože „super sport“ nebyl podle svého id nalezen, je vrácena hodnota -1. Při porušení integritních omezení je navracena hodnota -2. Povinný je pouze parametr T401Id. Zbylé parametry může být nahrazen hodnotou null, aby jeho hodnota zůstala nezměněna.

T402Sports

Tabulka druhů sportů podle pohlaví hráčů.

Jméno	Typ	Vlastnosti	Význam
T402Id	INT	Primární klíč	Jednoznačná identifikace
T401Id	INT	Cizí klíč (T401SuperSports)	K jakému „super sportu“ se

			vztahuje?
Title	VARCHAR(64)	Nenulový, délky alespoň 3 znaky	Jméno sportu, např. HOKEJ MUZI, HOKEJ ZENY, BASKETBAL MUZI, BASKETBAL ZENY...

Příklad:

T402Id	T401Id	title
21	11	HOKEJ MUZI
22	12	FOTBAL MUZI
23	13	BASKETBAL MUZI
24	14	HAZENA ZENY
25	14	HAZENA MUZI

Obrázek 33: Příklad uložení "super sportu"

Kdo je používá:

Sports Cribber (strana 15), *Businessman* (strana 34)

Indexy:

→ Přes T401Id

Přidružené funkce:

Funkce a parametry	Popis
PROCEDURE T402Insert (@T401Id INT , @title VARCHAR (255))	Procedura, která přidá nový sport. Jména parametrů odpovídají předchozí tabulce. Vrací id přidávaného záznamu. Pokud se operace nezdaří, je vrácena hodnota -1. Oba parametry jsou povinné.
PROCEDURE T402Update (@T402Id INT , @newT401Id INT , @newTitle VARCHAR (255))	Procedura upraví již vytvořený sport. Jména parametrů odpovídají jménům položek v předchozí tabulce. Vrací 0 při úspěchu. Pokud se procedura nezdaří, protože sport nebyl podle svého id nalezen, je vrácena hodnota -1. Při porušení integritních omezení je navracena hodnota -2. Povinný je pouze parametr T402Id. Zbylé parametry mohou být nahrazeny hodnotou null, aby jejich hodnota zůstala nezměněna.

T403Competitions

Tabulka soutěží.

Jméno	Typ	Vlastnosti	Význam
T403Id	INT	Primární klíč	Jednoznačná identifikace
T402Id	INT	Cizí klíč (T402Sports)	V jakém sportu se soutěží
Title	VARCHAR(64)	Nenulový, délky alespoň 3 znaky	Jméno soutěže, např. NHL

Dvojice T402Id, title je v rámci tabulky jedinečná.

Příklad:

T403Id	T402Id	title
21	21	AHL MUZI
42	21	ČESKÁ 1. HOKEJOVÁ LIGA MUZI
28	21	ČESKÁ EXTRALIGA MUZI
70	21	DÁNSKÁ HOKEJOVÁ LIGA MUZI
31	21	FINSKÁ HOKEJOVÁ LIGA MUZI
35	21	NĚMECKÁ HOKEJOVÁ LIGA MUZI
112	21	NĚMECKÝ HOKEJOVÝ POHÁR MUZI
24	21	NHL MUZI

Obrázek 34: Příklad uložení soutěží

Kdo je používá:

SportsDownload (strana 15), *Businessman* (strana 34)

Indexy:

→ Přes T402Id

Přidružené funkce:

Funkce a parametry	Popis
PROCEDURE T403Insert (@T402Id INT, @title VARCHAR(255))	Procedura, která přidá novou soutěž. Jména parametrů odpovídají předchozí tabulce. Vrací id přidávaného záznamu. Pokud se operace nezdaří, je vrácena hodnota -1. Oba parametry jsou povinné.

Funkce a parametry	Popis
<pre> PROCEDURE T403Update (@T403Id INT, @newT402Id INT, @newTitle VARCHAR(255)) </pre>	Procedura upraví již vytvořenou soutěž. Jména parametrů odpovídají jménům položek v předchozí tabulce. Vrací 0 při úspěchu. Pokud se procedura nezdaří, protože soutěž nebyla podle svého id nalezena, je vrácena hodnota -1. Při porušení integritních omezení je navrácena hodnota -2. Povinný je pouze parametr T403Id. Zbylé parametry můžou být nahrazeny hodnotou null, aby jejich hodnota zůstala nezměněna.

T404CompetitionYears

Tabulka jednotlivých ročníků soutěží.

Jméno	Typ	Vlastnosti	Význam
T404Id	INT	Primární klíč	Jednoznačná identifikace
T403Id	INT	Cizí klíč (T403Competitions)	Jaké soutěže se ročník týká
yearFrom	DATETIME	Nenulový, od 1.1.1990 dále	První rok soutěže (např. pokud je soutěž ročník 2002/2003, v tomto poli bude uloženo 2002). Z uloženího data se bere v potaz pouze uložení rok, zbytek data je ignorován.
yearTo	DATETIME	Nenulový, od 1.1.1990 dále	Druhý rok soutěže (např. pokud je soutěž ročník 2002/2003, v tomto poli bude uloženo 2003). Z uloženího data se bere v potaz pouze uložení rok, zbytek data je ignorován.

Trojice TT403Id, yearFrom, yearTo je v rámci tabulky jedinečná.

Příklad:

T404Id	T403Id	yearFrom	yearTo
1	1	1.1.2007 0:00:00	31.12.2008 0:00:00
314	1	1.1.2008 0:00:00	31.12.2009 0:00:00
2	2	1.1.2007 0:00:00	31.12.2008 0:00:00
235	2	1.1.2008 0:00:00	31.12.2009 0:00:00
3	3	1.1.2007 0:00:00	31.12.2008 0:00:00
216	3	1.1.2008 0:00:00	31.12.2009 0:00:00
4	4	1.1.2007 0:00:00	31.12.2008 0:00:00

Obrázek 35: Příklad uložení ročníků soutěží

Kdo je používá:

SportsDownload (strana 15), *Businessman* (strana 34)

Indexy:

→ Přes T403Id

Přidružené funkce:

Funkce a parametry	Popis
<pre>PROCEDURE T404Insert (@T403Id INT, @yearFrom DATETIME, @yearTo DATETIME)</pre>	Procedura, která přidá nový ročník soutěže. Jména parametrů odpovídají předchozí tabulce. Vrací id přidaného záznamu. Pokud se operace nezdaří, je vrácena hodnota -1. Všechny parametry jsou povinné.
<pre>PROCEDURE T404Update (@T404Id INT, @newT403Id INT, @newYearFrom DATETIME, @newYearTo DATETIME)</pre>	Procedura upraví již vytvořený ročník soutěže. Jména parametrů odpovídají jménům položek v předchozí tabulce. Vrací 0 při úspěchu. Pokud se procedura nezdaří, protože soutěž nebyla podle svého id nalezena, je vrácena hodnota -1. Při porušení integritních omezení je navracena hodnota -2. Povinný je pouze parametr T404Id. Zbylé parametry mohou být nahrazeny hodnotou null, aby jejich hodnota zůstala nezměněna.
<pre>CREATE TRIGGER T404ForInsertUpdate ON T404CompetitionYears FOR INSERT, UPDATE</pre>	Trigger, který kontroluje, zda zadané pole yearFrom je menší nebo rovno poli yearTo

T405Matches

Tabulka jednotlivých zápasů a jejich výsledků.

Jméno	Typ	Vlastnosti	Význam
T405Id	INT	Primární klíč	Jednoznačná identifikace
T404Id	INT	Cizí klíč (T404CompetitionYears), musí být zadáný	Ročník soutěže, do níž zápas spadá
T410IdHome	INT	Cizí klíč (T410Teams), nenulový	Tým, který se zápasu účastnil
T410IdVisit	INT	Cizí klíč (T410Teams), nenulový	Tým, který se zápasu účastnil
round	INT	Nenulový, větší, než 0	Číslo kola soutěže
time	DATETIME	Nenulovým od 1.1.1990 dále	Datum, kdy se zápas hrál, nebo hrát teprve bude
homeScore	INT	Buď je nulové (null nebo 0), nebo má kladnou hodnotu	Skóre, které získal první (domácí) tým
visitScore	INT	Buď je nulové (null nebo 0), nebo má kladnou hodnotu	Skóre, které získal druhý (hostující) tým
prolongation	CHAR(1)	Hodnota je null, prázdný řetězec, nebo znak P	Příznak, zda byl zápas prodlužován
visitScore	INT	Buď je nulové (null nebo 0), nebo má kladnou hodnotu	Skóre, které získal druhý (hostující) tým

Dvojice T404Id, round je v rámci tabulky jedinečná. Buď jsou všechny hodnoty homeScore, visitScore a prolongation null, nebo není ani jedna z nich.

Příklad:

T405Id	T404Id	T410IdHome	T410IdVisit	round	time	homeScore	visitScore	prolongation
2007	31	42	43	0	1/25/2009 12:00:00 AM	1	5	
2008	32	44	45	0	1/25/2009 12:00:00 AM	NULL	NULL	NULL
2009	32	46	47	0	1/25/2009 12:00:00 AM	1	3	
2010	32	48	49	0	1/25/2009 12:00:00 AM	0	2	
2011	32	50	51	0	1/20/2009 4:00:00 PM	NULL	NULL	NULL
2012	32	52	53	0	1/25/2009 12:00:00 AM	3	1	
2013	32	54	55	0	1/20/2009 4:00:00 PM	NULL	NULL	NULL
2014	32	56	57	0	1/25/2009 12:00:00 AM	NULL	NULL	NULL
2015	32	58	59	0	1/25/2009 12:00:00 AM	1	1	
2016	32	60	61	0	1/25/2009 12:00:00 AM	2	2	

Obrázek 36: Příklad uložení ročníků soutěží

Pro příklad analyzujeme první řádek – z hodnoty pole T404Id a příslušné tabulky T404competitionYears víme, že se jedná o ročník 2008/2009 soutěže AHL muži (tabulka T403Competitions). Z hodnot polí T401IdHome a T401IdVisit (a příslušné tabulky T410Teams) je zřejmé, že hrály týmy San Antonio Rampage (id 42) a Lake Erie Monsters (id 43). Jednalo se o první kolo v rámci soutěže a San Antonio prohráli s Monsters 1:5 v běžné hrací době.

Kdo je používá:

SportsDownload (strana 15), *Businessman* (strana 34), *Occupant* (strana 49)

Indexy:

→ Přes T404Id, T410IdHome a T410IdVisit

Přidružené funkce:

Funkce a parametry	Popis
<pre>PROCEDURE T405Insert (@T404Id INT, @T410IdHome INT, @T410IdVisit INT, @round INT, @date DATETIME)</pre>	Procedura, která přidá nový zápas. Jména parametrů odpovídají předchozí tabulce. Vrací id přidaného záznamu. Pokud se operace nezdaří, je vrácena hodnota -1. Všechny parametry jsou povinné.
<pre>PROCEDURE T405Update (@T405Id INT, @newT404Id INT, @newT410IdHome INT, @newT410IdVisit INT, @newRound INT, @newDate DATETIME)</pre>	Procedura upraví již vytvořený zápas. Jména parametrů odpovídají jménům položek v předchozí tabulce. Vrací 0 při úspěchu. Pokud se procedura nezdaří, protože zápas nebyl podle svého id nalezen, je vrácena hodnota -1. Při porušení integritních omezení je navracena hodnota -2. Povinný je pouze parametr T405Id. Zbylé parametry můžou být nahrazeny hodnotou null, aby jejich hodnota zůstala nezměněna.
<pre>PROCEDURE T405UpdateScore (@T405Id INT, @newHomeScore INT, @newVisitScore INT, @newProlongation CHAR(1))</pre>	Procedura, jejímž smyslem je aktualizovat hodnoty skóre zápasu. Jména parametrů odpovídají jménům položek v předchozí tabulce. Vrací 0 při úspěchu. Pokud se procedura nezdaří, protože zápas nebyl podle svého id nalezen, je vrácena hodnota -1. Při porušení integritních omezení je navracena hodnota -2. Povinné jsou všechny parametry.

T406Opportunities

Tabulka zaznamenávající průměrné hodnoty kurzů vypsanych sázkovými kanceláři na jednotlivé možnosti výsledku zápasu.

Jméno	Typ	Vlastnosti	Význam
T406Id	INT	Primární klíč	Jednoznačná identifikace

T405Id	INT	Cizí klíč (T405Matches), povinně zadaný	Zápas, kterého se kurz týká
type	INT	Vždy zadaný, má hodnotu z intervalu <1, 5>	Výsledek zápasu: 1 ... výhra domácích 2 ... ne-prohra domácích 3 ... remíza 4 ... ne-prohra hostů 5 ... výhra hostů
rate	DECIMAL(8, 3)	Vždy zadaný, větší než 1	Vypsáný kurz

Dvojice T405Id, type je v rámci tabulky jedinečná.

Příklad:

T406Id	T405Id	type	rate
6905	2007	1	1.973
6906	2007	2	1.320
6907	2007	3	3.985
6908	2007	4	1.645
6909	2007	5	2.801
6910	2009	1	3.049

Obrázek 37: Příklad uložení kurzů vypsáných na zápasy sázkovými kanceláři

Vidíme, že na zápas s id 2007 (detaily zápasu jsou analyzované v příkladu u obrázku 36 na straně 90) byly vypsány tyto kurzy:

- na výhru domácích 1,973
- na výhru domácích nebo remízu 1,320
- na remízu 3,985
- na remízu nebo výhru hostů 1,645
- a na výhru hostů 2,801

Kdo je používá:

SportsDownload (strana 15), *Businessman* (strana 34), *Occupant* (strana 49), *Killguard* (strana 55)

Indexy:

- ➔ Přes T405Id

Přidružené funkce:

Funkce a parametry	Popis
<pre>CREATE PROCEDURE T406Insert (@T405Id INT, @type VARCHAR(2), @rate INT)</pre>	Procedura, která přidá kurz vypsany na jednu možnost výsledku jednoho zápasu. Jména parametrů odpovídají předchozí tabulce. Vrací id přidaného záznamu. Pokud se operace nezdaří, je vrácena hodnota -1. Všechny parametry jsou povinné.
<pre>PROCEDURE T406Update (@T406Id INT, @newT405Id INT, @newType VARCHAR(2), @newRate INT)</pre>	Procedura upraví již vložený vypsany kurz. Jména parametrů odpovídají jménům položek v předchozí tabulce. Vrací 0 při úspěchu. Pokud se procedura nezdaří, protože zápas nebyl podle svého id nalezen, je vrácena hodnota -1. Při porušení integritních omezení je navracena hodnota -2. Povinný je pouze parametr T406Id. Zbylé parametry můžou být nahrazeny hodnotou null, aby jejich hodnota zůstala nezměněna.

T410Teams

Tabulka pro uložení týmů.

Jméno	Typ	Vlastnosti	Význam
T410Id	INT	Primární klíč	Jednoznačná identifikace
T402Id	INT	Cizí klíč (T402Sports), povinně zadaný	Sport, ve kterém tým závodí
title	VARCHAR(255)	Vždy zadaný, délky alespoň 3 znaky, velkými písmeny	Jméno týmu
rate	DECIMAL(8, 3)	Vždy zadaný, větší než 1	Vypsany kurz

Dvojice T402Id, title je v rámci tabulky jedinečná.

Příklad:

T410Id	T402Id	title
42	21	HOKEJ SAN ANTONIO RAMPAGE
43	21	HOKEJ LAKE ERIE MONSTERS
44	22	FOTBAL CAMBRIDGE
45	22	FOTBAL HISTON
46	22	FOTBAL YORK
47	22	FOTBAL BURTON
48	22	FOTBAL LEWES

Obrázek 38: Příklad uložení jmen týmů

Týmy San Antonio Rampage a Lake Erie Monsters se věnují mužskému hokeji (sport s id 21). Zbýlé zobrazené týmy mužskému fotbalu (sport s id 22).

Kdo je používá:

SportsDownload (strana 15)

Indexy:

→ Přes T402Id

Přidružené funkce:

Funkce a parametry	Popis
<pre>PROCEDURE T410Insert (@T402Id INT, @title VARCHAR(255))</pre>	Procedura, která přidá nový tým. Jediný a povinný parametr je jméno týmu. Vrací id přidaného záznamu. Pokud se operace nezdaří, je vrácena hodnota -1.
<pre>PROCEDURE T410Update (@T410Id INT, @newT402Id INT, @newTitle VARCHAR(255))</pre>	Procedura upraví jméno již uloženého týmu nebo sportovní odvětví, kterému se věnuje. Jména parametrů odpovídají jménům položek v předchozí tabulce. Vrací 0 při úspěchu. Pokud se procedura nezdaří, protože tým nebyl podle svého id nalezen, je vrácena hodnota -1. Při porušení integritních omezení je navracena hodnota -2. Povinný je pouze parametr T410Id. Zbýlé parametry můžou být nahrazeny hodnotou null, aby jejich hodnota zůstala nezměněna.

T420OpportunitiesBet

Tabulka pro uložení tipů výsledků jednotlivých zápasů od jednotlivých instancí. Pokud je zrušena instance (smazána z databáze), jsou smazány i její tipy – odpovídající řádky této tabulky.

Jméno	Typ	Vlastnosti	Význam
T420Id	INT	Primární klíč	Jednoznačná identifikace
T303Id	INT	Cizí klíč (T303InvestInstance), povinně zadáný	Instance, která je autorem tipu
T405Id	INT	Cizí klíč (T405Matches), povinně zadáný	Zápas, kterého se tip týká
type	INT	Vždy zadáný, z intervalu <1, 5>	Výsledek zápasu: 1 ... výhra domácích 2 ... ne-prohra domácích 3 ... remíza 4 ... ne-prohra hostů 5 ... výhra hostů
rate	DECIMAL(8, 3)	Povinně zadáný, větší než 1	Odhad kurzu

Dvojice T420Id, type je v rámci tabulky jedinečná.

Příklad:

T420Id	T303Id	T405Id	type	rate
1	101	4019	1	6,735
2	101	4019	2	2,422
3	101	4019	3	3,935
4	101	4019	4	1,167
5	101	4019	5	1,807
6	101	4692	1	3,379
7	101	4692	2	1,578
8	101	4692	3	3,749
9	101	4692	4	1,880
10	101	4692	5	4,662
11	101	4757	1	2,412
12	101	4757	2	1,349

Obrázek 39: Příklad

Kdo je používá:

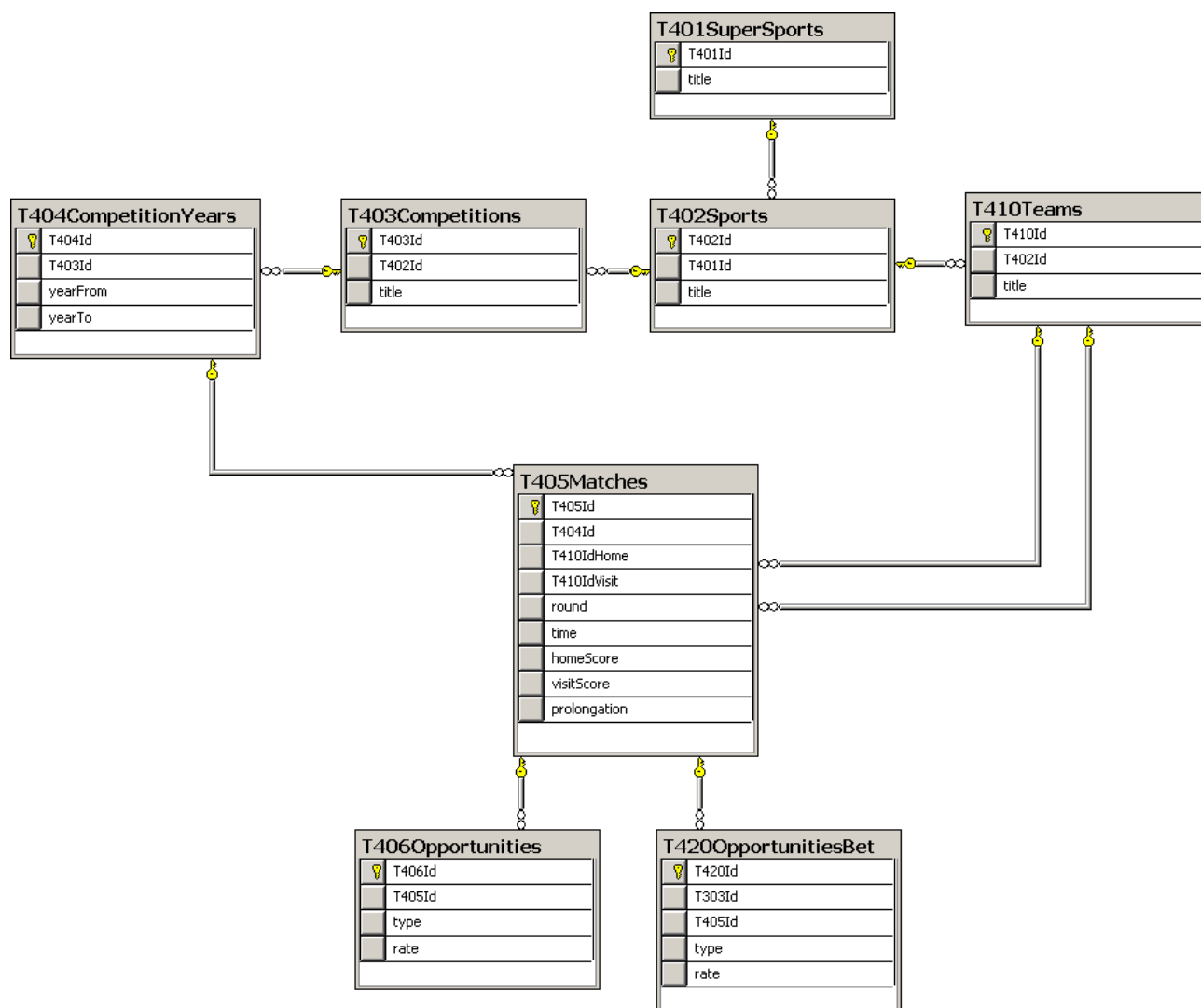
SportsDownload (strana 15), *Businessman* (strana 34)

Indexy:

→ žádné

Přidružené funkce:

Funkce a parametry	Popis
<pre>PROCEDURE T420InsertOrUpdate (@T303Id INT, @T405Id INT, @type INT, @rate DECIMAL(8, 3))</pre>	Procedura, která přidá nový odhad vypsaneho kurzu. Jména parametrů odpovídají popisu v předchozí tabulce. Vrací id přidaného záznamu. Pokud se operace upravení nezdaří, je vrácena hodnota -2 a když se nepodaří vložit, vrací -1. Všechny parametry jsou povinné.
<pre>PROCEDURE T410Update (@T410Id INT, @newT402Id INT, @newTitle VARCHAR(255))</pre>	Procedura upraví jméno již uloženého týmu nebo sportovní odvětví, kterému se věnuje. Jména parametrů odpovídají jménům položek v předchozí tabulce. Vrací 0 při úspěchu. Pokud se procedura nezdaří, protože tým nebyl podle svého id nalezen, je vrácena hodnota -1. Při porušení integritních omezení je navracena hodnota -2. Povinný je pouze parametr T410Id. Zbylé parametry můžou být nahrazeny hodnotou null, aby jejich hodnota zůstala nezměněna.



Obrázek 40

Další funkce

Následující dvě funkce slouží pro snazší ukládání nebo modifikace dat v databázi – není třeba vše postupně ručně ukládat nebo měnit (nejdříve vytvořit „super sport“, potom sport, týmy...), tyto dvě funkce to za nás udělají.

T400InsertedMatchId

Funkce, která se používá pro uložení zápasu před jeho odehráním (a nebo po odehrání zápasu, ale neukládá výsledky – to je práce další funkce). Zadané parametry jsou vloženy do databáze. Hodnoty, které již v databázi existovaly, nejsou duplikovány. Mezi parametry je vytvořena (zachována) logická spojitost – předpokládá se, že všechny parametry se týkají jednoho zápasu.

Hlavičku funkce ukazuje text 22, parametry jsou popsány v tabulce následující níže.

Text 22: Hlavička funkce pro jednoduché ukládání dat do databáze nebo jejich modifikaci (data týkající se zápasu)

```
PROCEDURE T400InsertedMatchId
(
    @superSport VARCHAR(64),
    @sport VARCHAR(64),
    @competition VARCHAR(64),
    @yearFrom DATETIME,
    @yearTo DATETIME,
    @homeTeam VARCHAR(255),
    @visitTeam VARCHAR(255),
    @round INT,
    @time DATETIME
)
```

Parametr	Význam	Příklad
superSport	Jméno <i>super sportu</i>	FOTBAL
sport	Jméno <i>sportu</i>	FOTBAL MUZI
competition	Jméno <i>soutěže</i>	1. ANGLICKÁ LIGA MUZI
yearFrom	První rok, do kterého ročník soutěže spadá	1/1/2008 12:00:00 AM
yearTo	Druhý rok, do kterého ročník soutěže spadá	1/1/2009 12:00:00 AM
homeTeam	Tým domácích (jeden z týmů účastníků se zápasu)	FOTBAL ASTON VILLA
visitTeam	Tým hostů (druhý z týmů účastníků se zápasu)	FOTBAL CHELSEA
round	Kolo soutěže	0
time	Datum a čas zápasu	2009-02-21 13:45:00.000

Všechny parametry jsou povinné. V případě úspěchu vrací kladné číslo zápasu (id). Pokud nastala chyba, je návratová hodnota záporná:

Navracená hodnota	Význam
x, x>0	Úspěch, x je <i>id</i> zápasu
-1	Nepodařilo se vložit <i>super sport</i>
-2	Nepodařilo se vložit <i>sport</i>
-3	Nepodařilo se vložit tým domácích
-4	Nepodařilo se vložit tým hostů
-5	Nepodařilo se vložit soutěž
-6	Nepodařilo se vložit ročník soutěže
-7	Nepodařilo se vložit zápas
-8	Nepodařilo se upravit zápas

Kdo je používá:

SportsDownload (strana 15), *Businessman* (strana 34)

T400InsertedMatchScore

Funkce, která se používá pro uložení výsledků zápasu a vypsaných kurzů po jeho odehrání. Zadané parametry jsou vloženy do databáze. Hodnoty, které již v databázi existovaly, nejsou duplikovány. Pokud je parametr již v databázi uložen, ale s jinou hodnotou, je jeho hodnota změněna. Mezi parametry je vytvořena (zachována) logická spojitost – předpokládá se, že všechny parametry se týkají jednoho zápasu.

Hlavičku funkce ukazuje text 23, parametry jsou popsány v tabulce následující níže.

Text 23: Hlavička funkce pro jednoduché ukládání dat do databáze nebo jejich modifikaci (data týkající se skóre zápasu)

```

PROCEDURE T400InsertedMatchScore
(
    @T405Id INT,
    @homeScore INT,
    @visitScore INT,
    @prolongation CHAR(1),
    @rate1 DECIMAL(8,3),
    @rate2 DECIMAL(8,3),
    @rate3 DECIMAL(8,3),
    @rate4 DECIMAL(8,3),
    @rate5 DECIMAL(8,3)
)

```

Parametr	Význam	Příklad
T405Id	Id dotčeného zápasu	10331
homeScore	Skóre domácích	0
visitScore	Skóre hostů	1
prolongation	Příznak, zda byl zápas prodlužován	<i>prázdný řetězec</i> (nebo P)
rate1	Kurz vypsaný na výhru domácích	3.199

Parametr	Význam	Příklad
rate2	Kurz vypsany na výhru domácích nebo remízu	1.599
rate3	Kurz vypsany na remízu	3.197
rate4	Kurz vypsany na remízu nebo výhru hostů	3.303
rate4	Kurz vypsany na výhru hostů	2.199

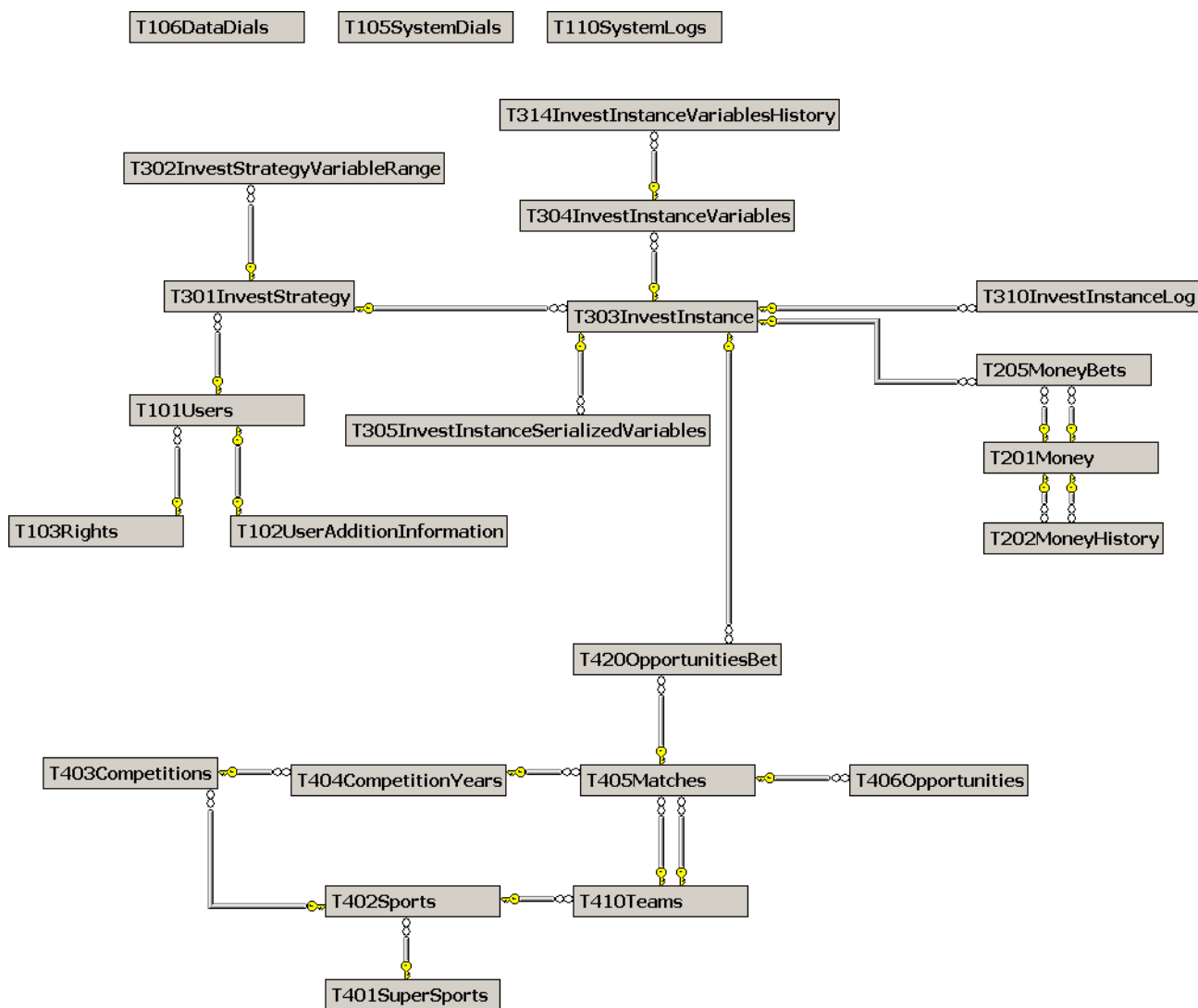
Všechny parametry jsou povinné. V případě úspěchu vrací nulu. Pokud nastala chyba, je návratová hodnota záporná. Pokud nastalo chyb více, je vrácena ta s nejnižší hodnotou.

Navracená hodnota	Význam
0	Úspěch
-1	Špatně zadané parametry (prázdné první čtyři parametry)
-2	Nepodařilo se aktualizovat/vložit skóre zápasu
-3	Nepodařilo se vložit 1. kurz
-4	Nepodařilo se aktualizovat 1. kurz
-5	Nepodařilo se vložit 2. kurz
-6	Nepodařilo se aktualizovat 2. kurz
-7	Nepodařilo se vložit 3. kurz
-8	Nepodařilo se aktualizovat 3. kurz
-9	Nepodařilo se vložit 4. kurz
-10	Nepodařilo se aktualizovat 4. kurz
-11	Nepodařilo se vložit 5. kurz
-12	Nepodařilo se aktualizovat 5. kurz

Kdo je používá:

SportsDownload (strana 15), *Businessman* (strana 34)

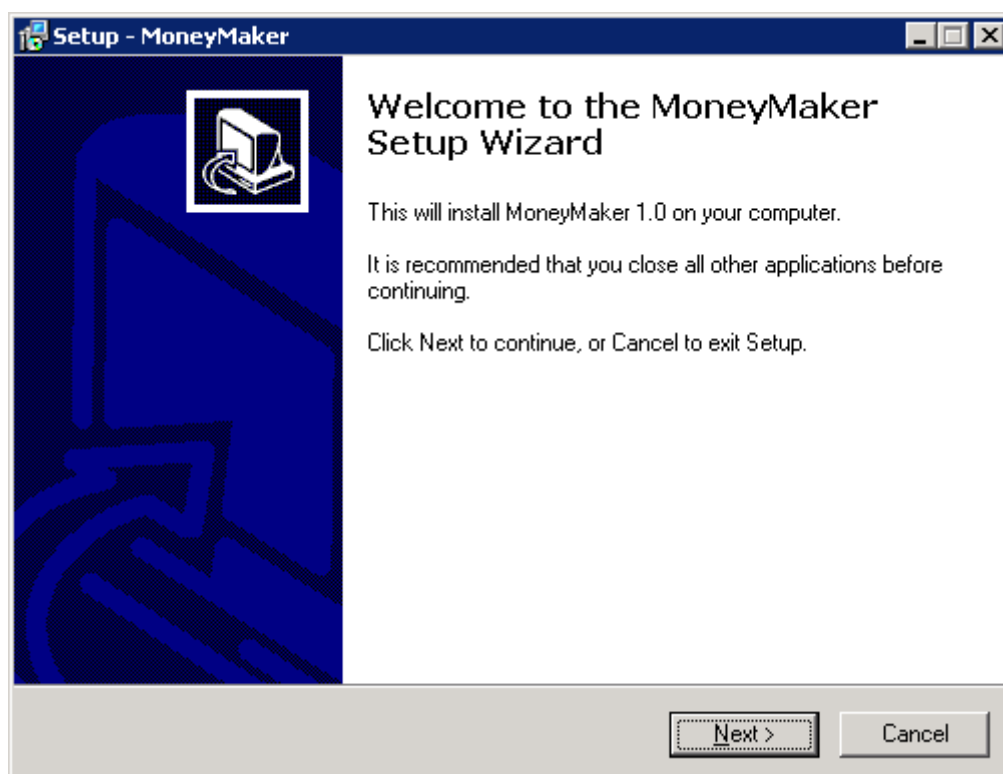
Schéma databáze jako celku



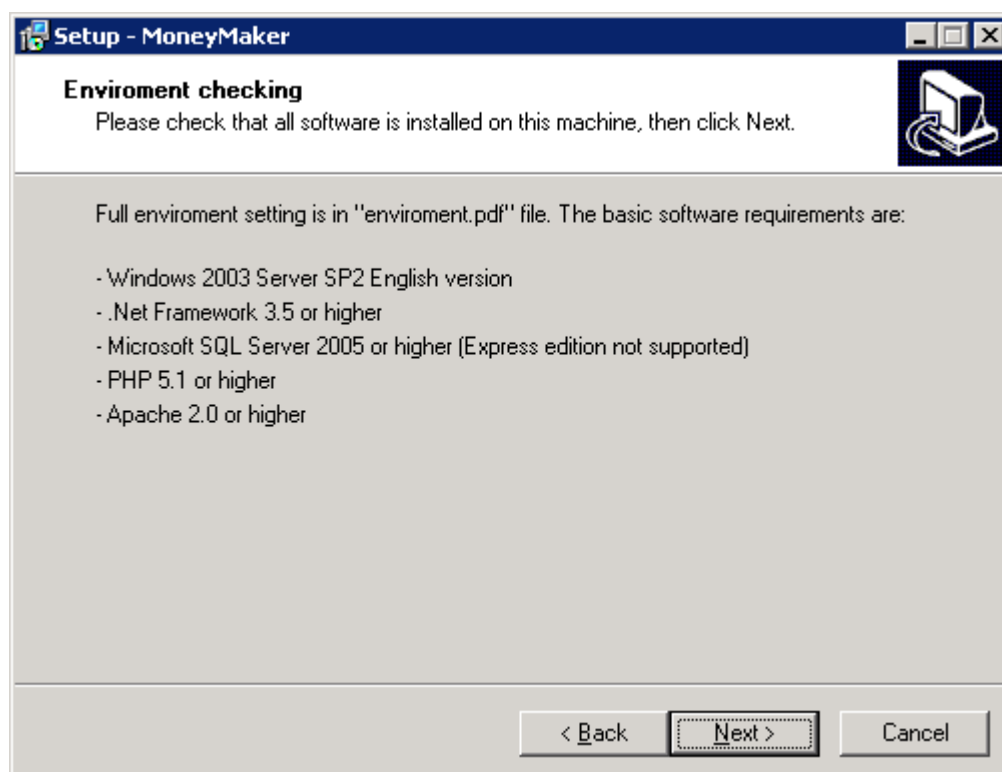
Obrázek 41

Instalace

Spustíte instalační program `server.exe`.

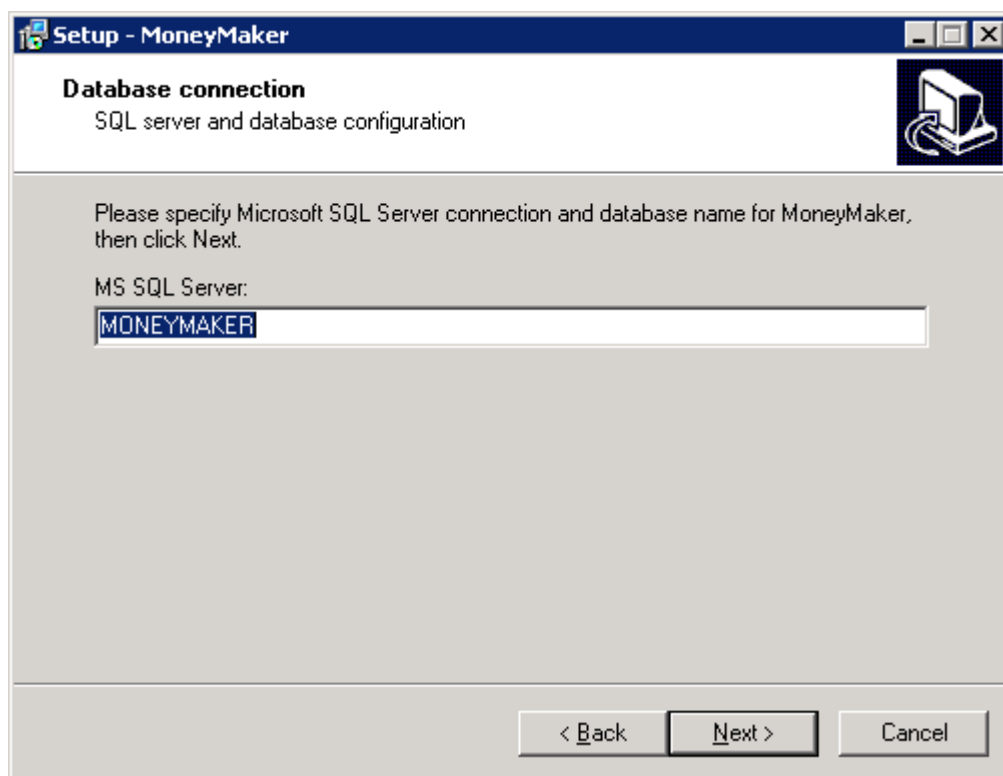


Obrázek 42



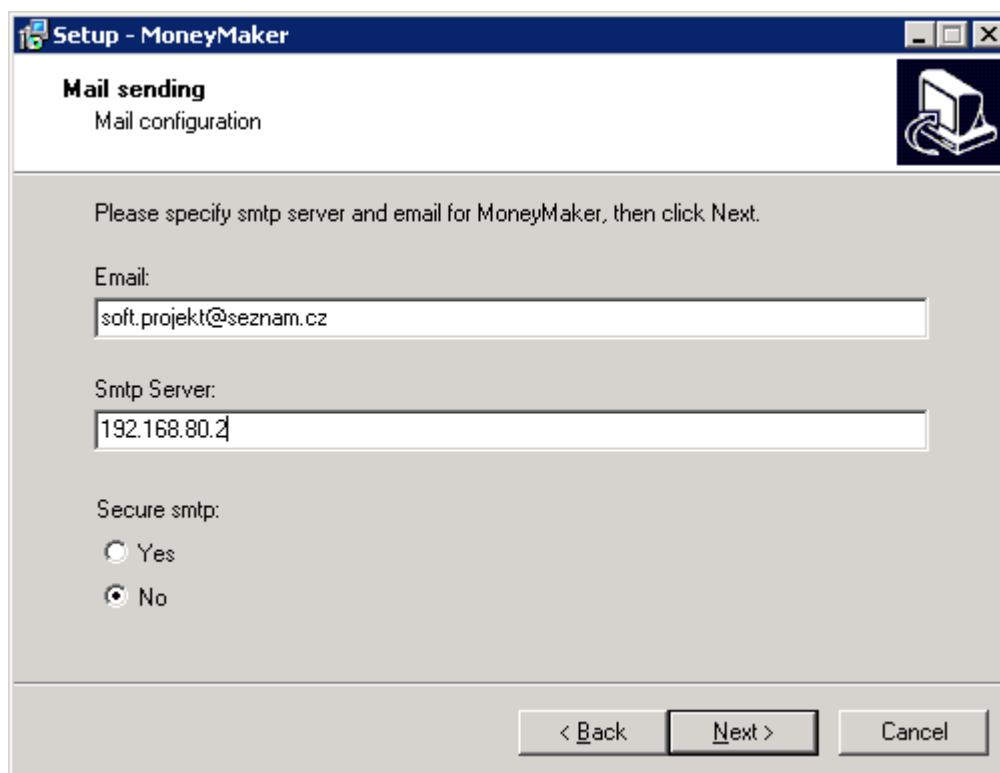
Obrázek 43

Než budete v instalaci pokračovat, **je nutné mít nainstalované některé další programy** – jejich seznam je na obrázku 43. Instalační program nekontroluje, zda je máte nainstalované, bude pokračovat dál, i když k dispozici nejsou, ale instalace se nezdaří (některé součásti nebudou moci být nainstalovány a celý systém proto nebude fungovat).



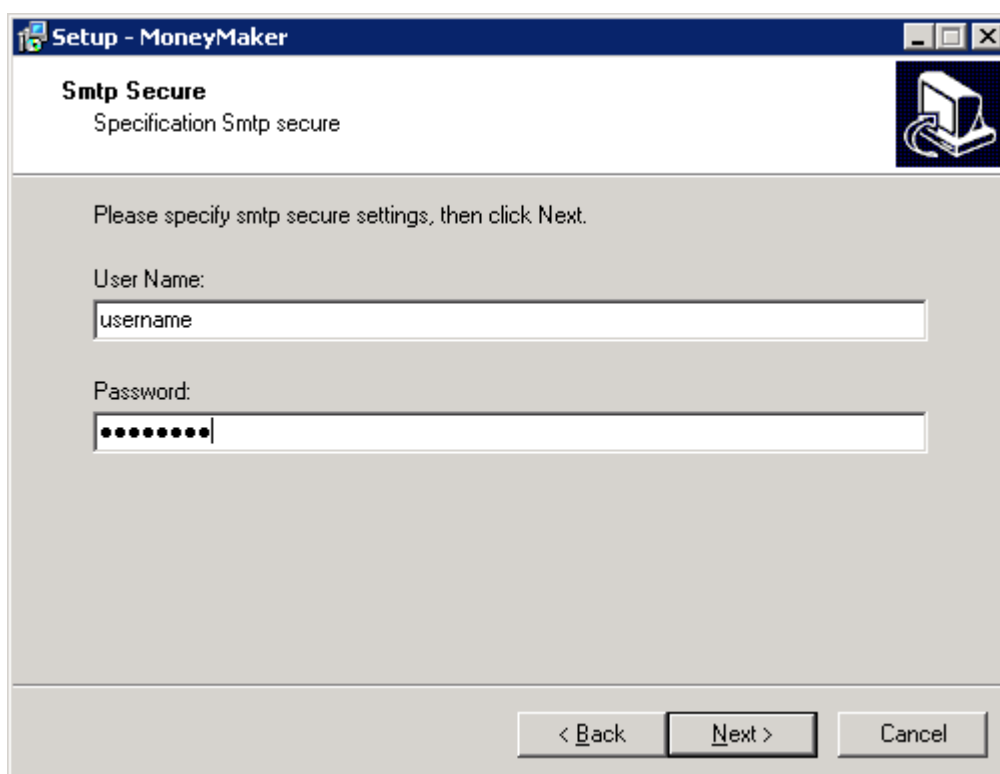
Obrázek 44

Zadejte jméno databáze (a spojení), kterou má systém využívat.



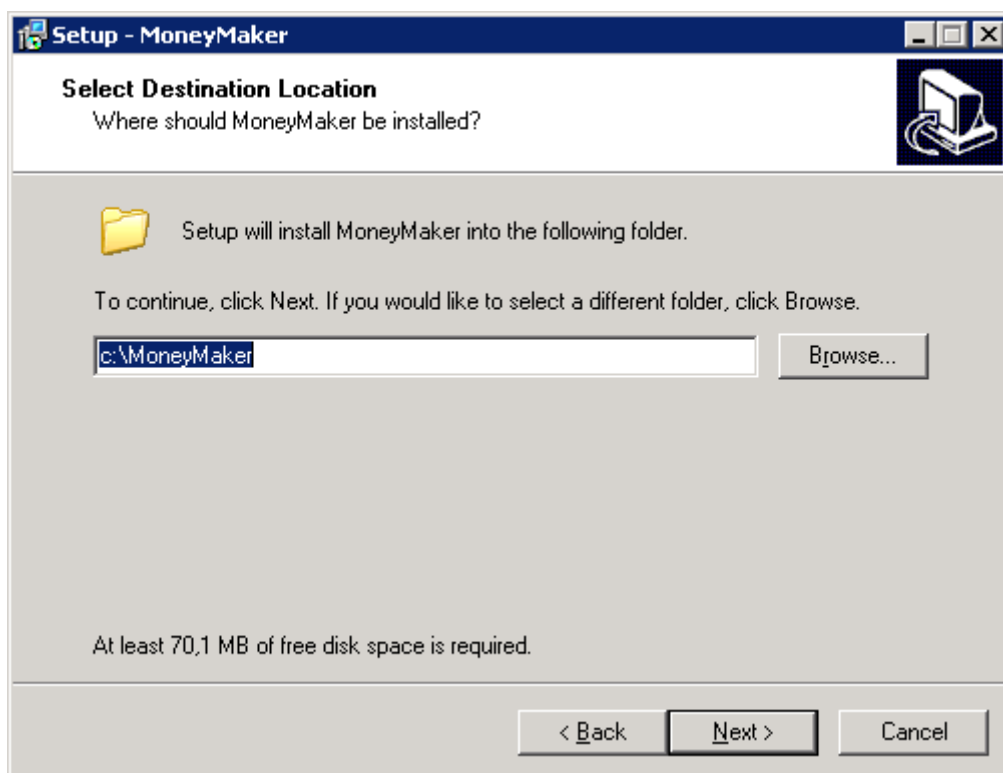
Obrázek 45

Zadejte email odesílatele (nikoli příjemce) chybových zpráv a adresu SMTP serveru, který bude pro tyto účely používán. Zaškrtněte, zda server vyžaduje autentizaci uživatele.



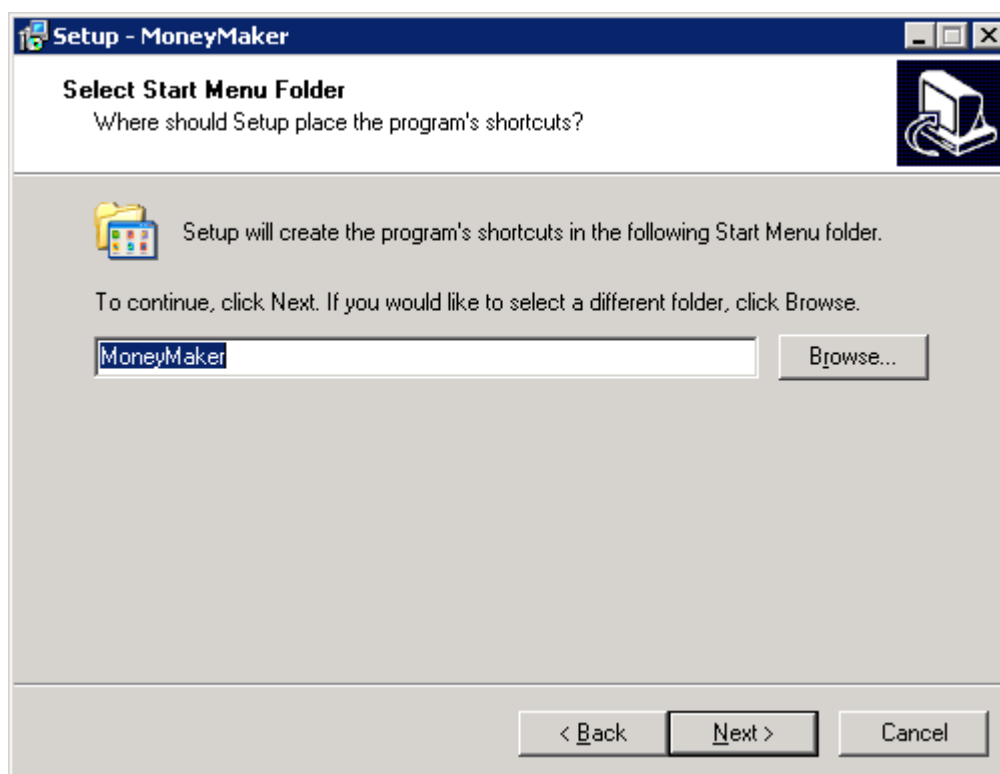
Obrázek 46

Pokud jste zadali, že server vyžaduje autentizaci, budete vyzváni k zadání příslušných údajů. Pokud ne, bude dialog na obrázku 46 přeskóčen.



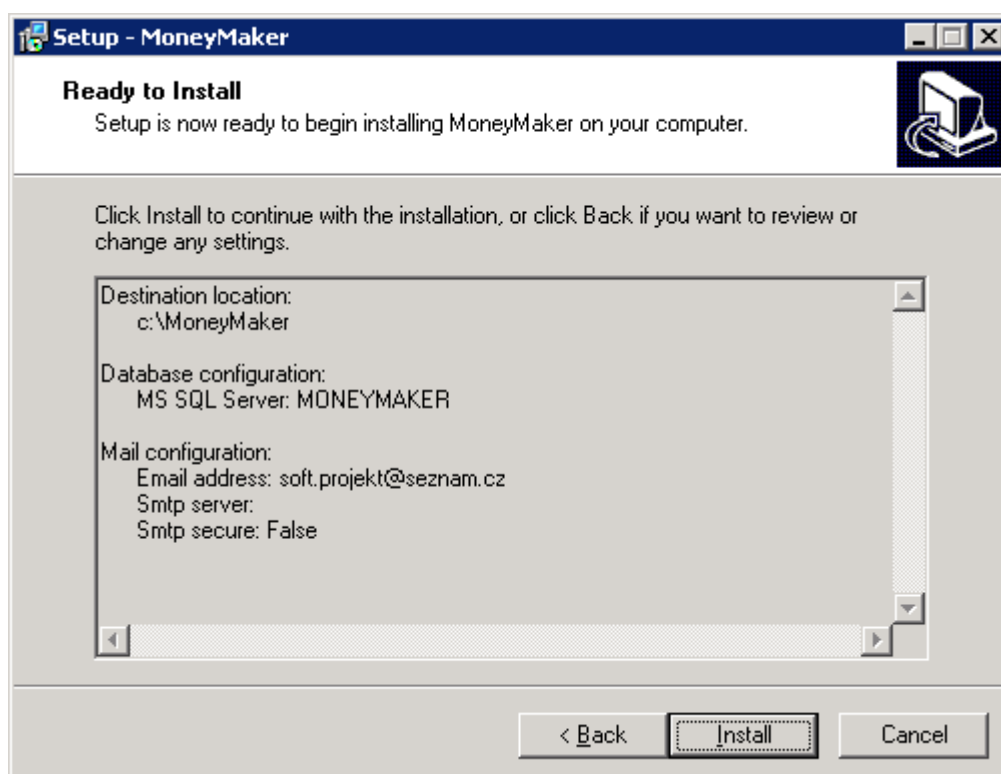
Obrázek 47

Zadejte cestu, kam má být systém *MoneyMaker* nainstalován.



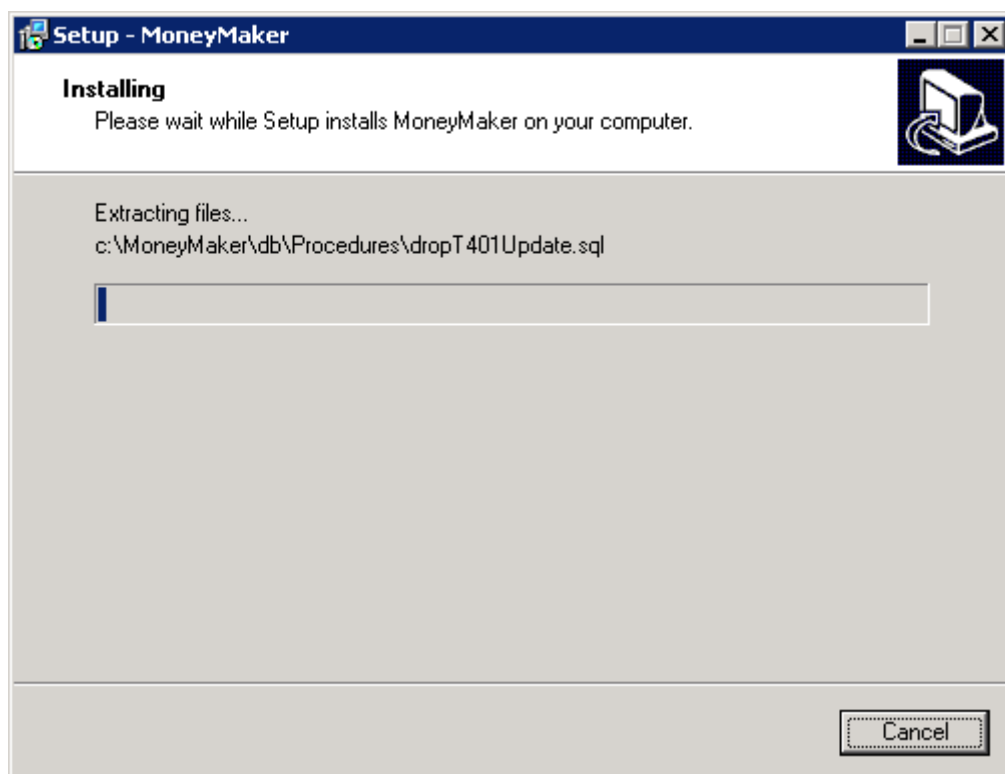
Obrázek 48

Zadejte jméno záložky, která se vytvoří v menu *Start* Vašeho systému.



Obrázek 49

Dialog na obrázku 49 je pouze informační, shrnuje zadané nastavení. Zkontrolujte, zda je vše tak, jak jste chtěli a klikněte na tlačítko *Install*.



Obrázek 50

Spustí se instalace a budete informováni o jejím průběhu.

```

C:\ Select SQLCMD
===== Info =====
SERVERNAME="MONEYMAKER"
===== Database =====

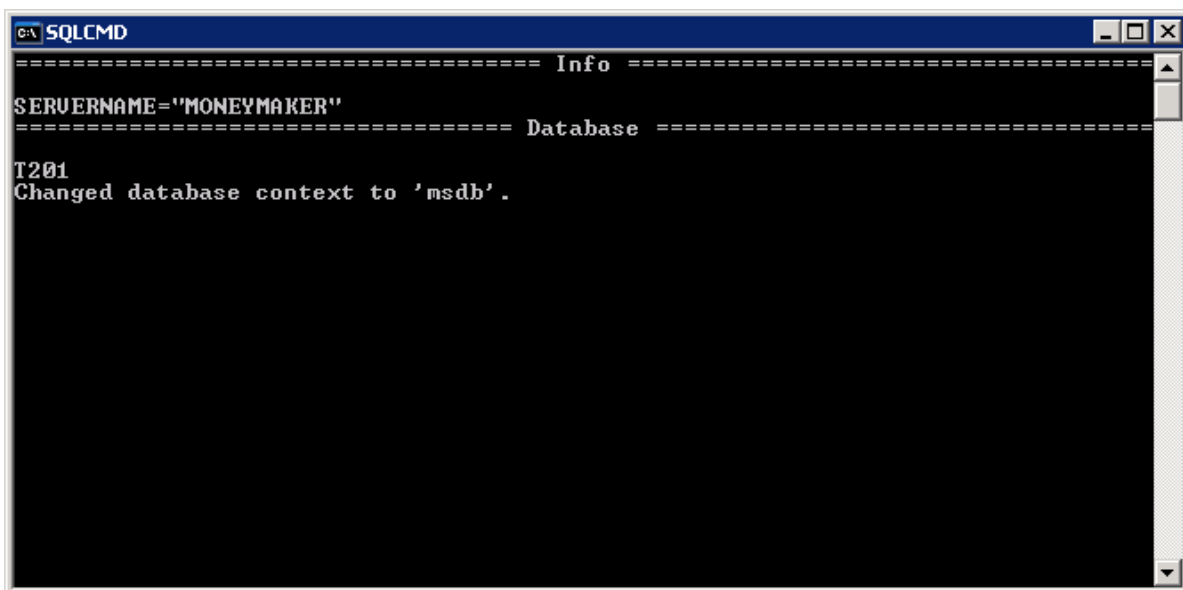
Creating new database MM
... script has done
Creating new tables
Changed database context to 'MMDB'.
... script has done
Initializing DBMM <fill data>
Changed database context to 'MMDB'.

<1 rows affected>
... script has done
MM
Changed database context to 'MMDB'.
Changed database context to 'MMDB'.
... scripts have done
Ti01
Changed database context to 'MMDB'.
Changed database context to 'MMDB'.
Changed database context to 'MMDB'.
Changed database context to 'MMDB'.
Changed database context to 'MMDB'.

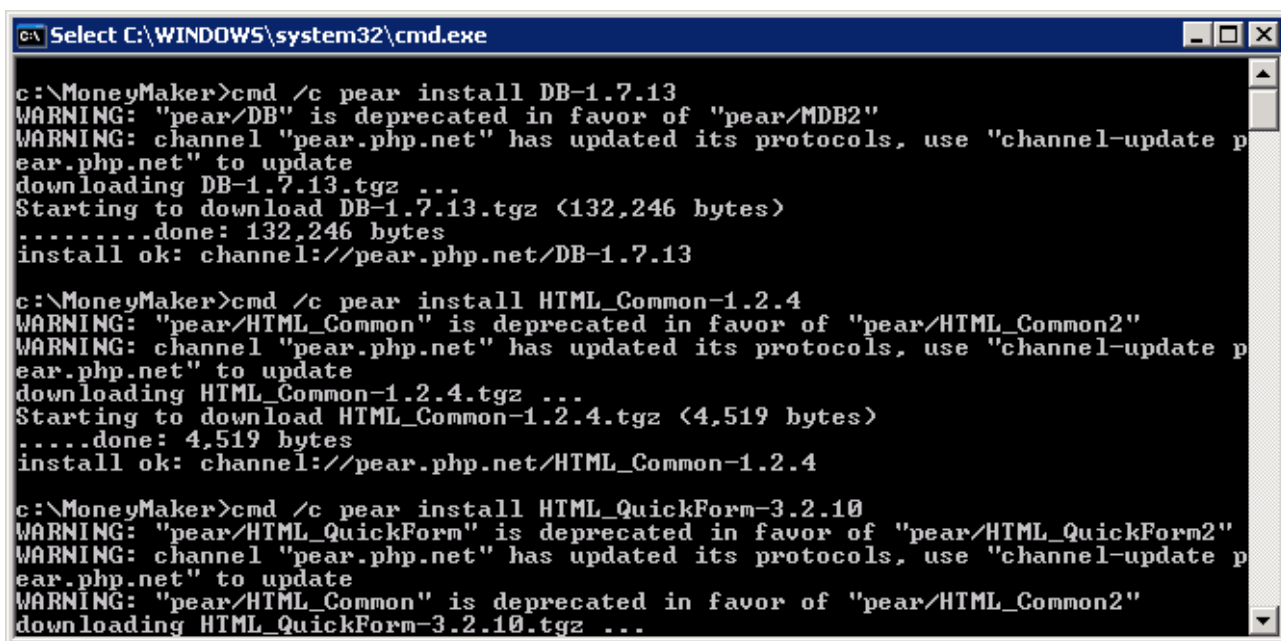
```

Obrázek 51

Jako první se bude instalovat obsah databáze (tabulky atd.).

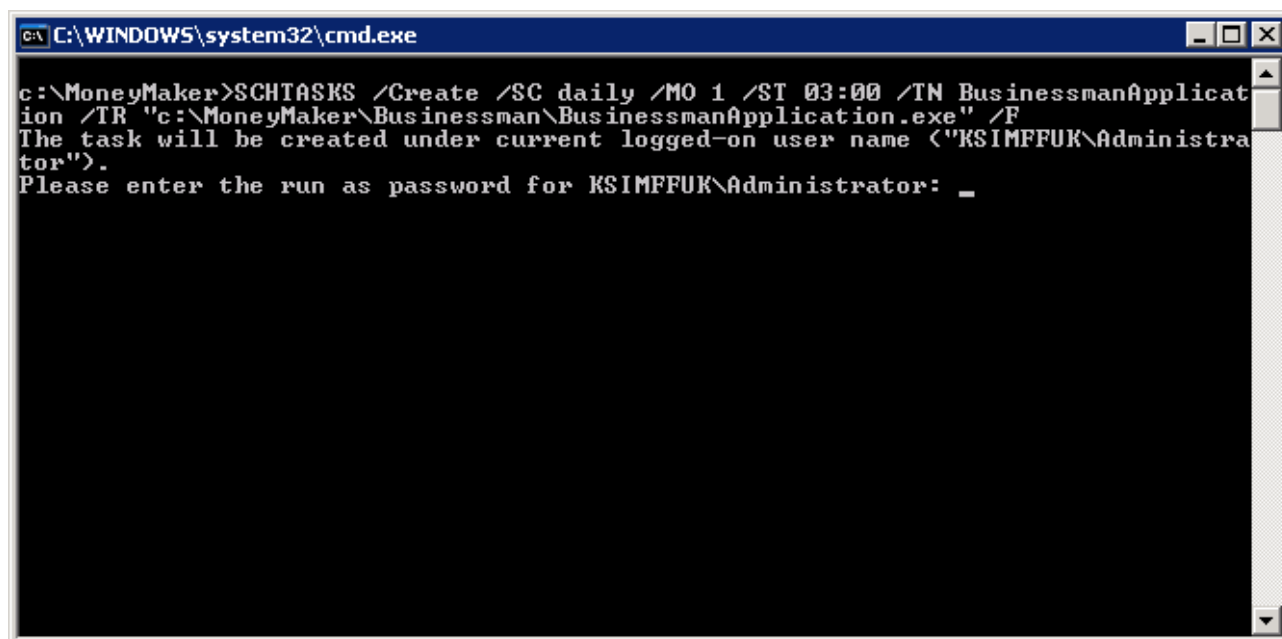


Obrázek 52



Obrázek 53

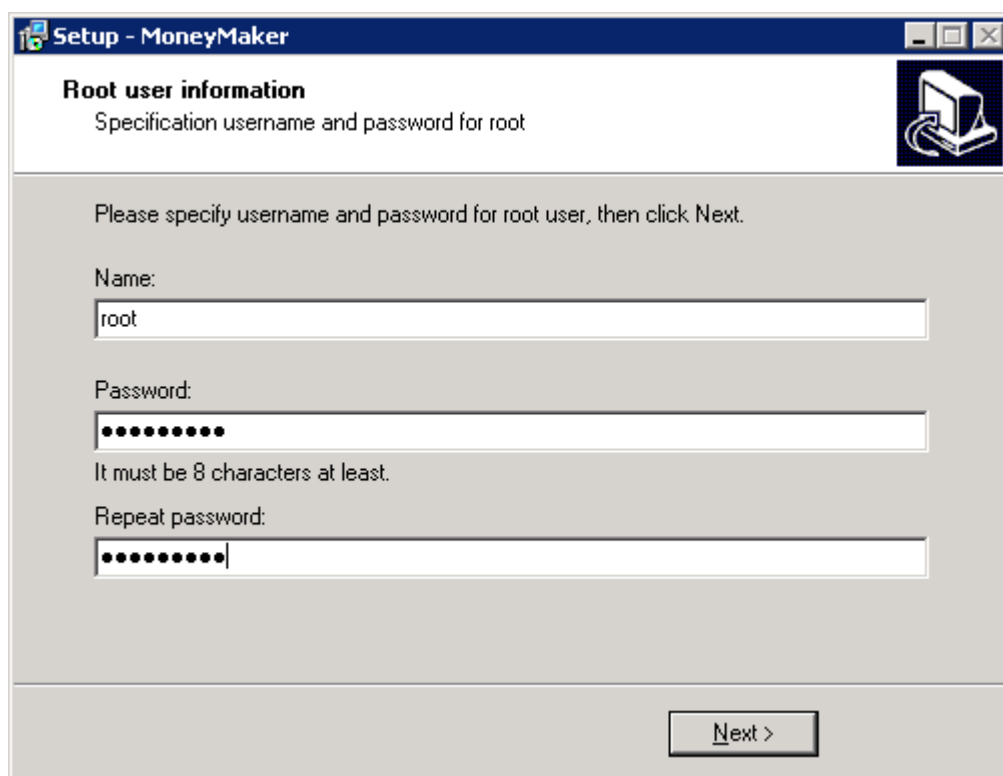
Následuje instalace *PEAR*.



Obrázek 54

Na konec jsou do systému nahrány pravidelně spouštěné úlohy*. Tyto úlohy se budou spouštět s právy administrátora, tedy budete vyzváni k zadání jeho hesla. Úlohy jsou čtyři.

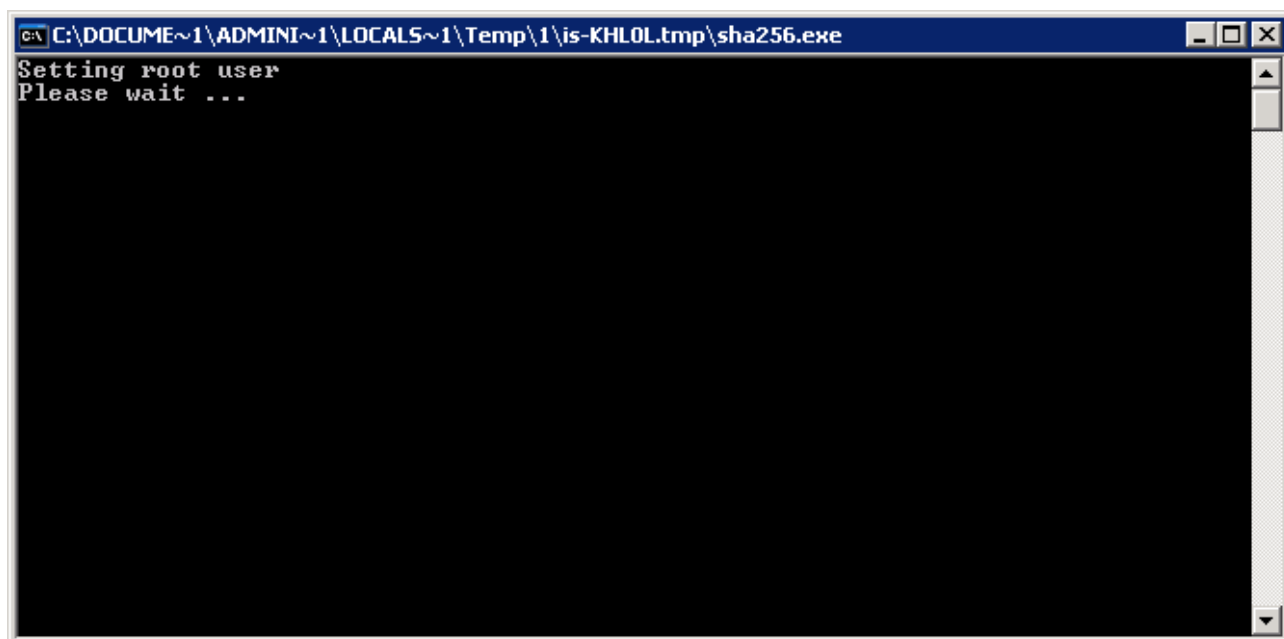
* Pokud instalujete MoneyMaker na jiný operační systém, než Windows 2003 ENG, budete muset tuto část provést ručně. Spouštějte všechny úlohy (BusinessmanApplication, Cribber – money, Cribber – sports a Killguard) denně, ideálně v nočních hodinách, kdy předpokládáte malý provoz na serveru.



Obrázek 55

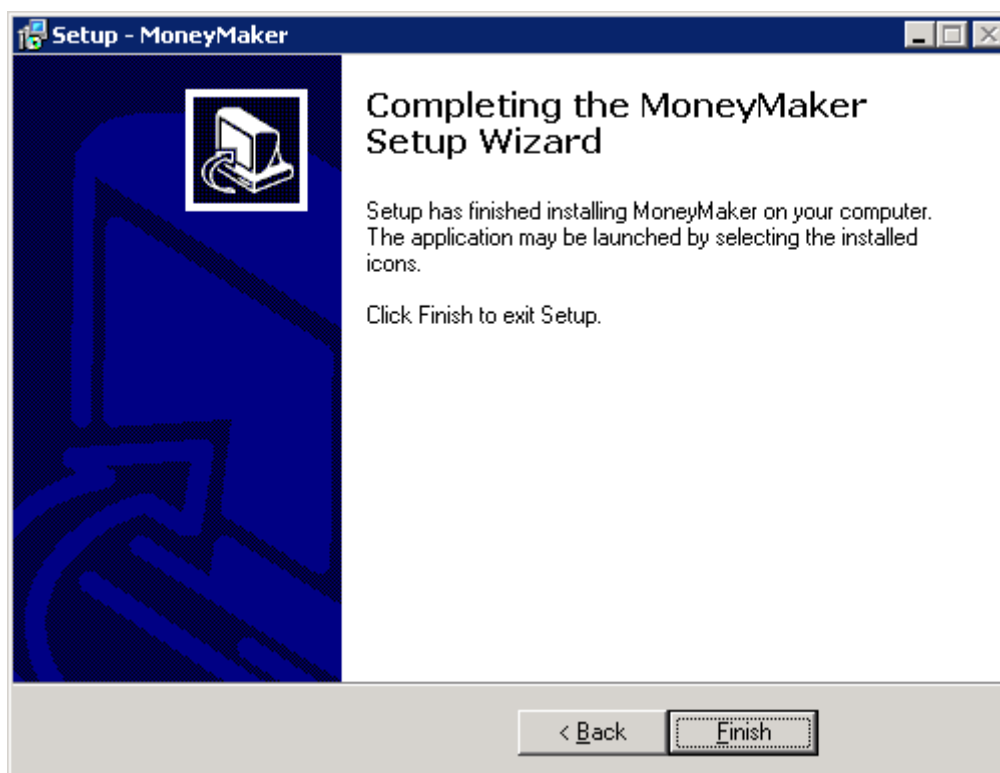
Nyní je třeba vytvořit administrátora systému *MoneyMaker*. Zadejte jeho uživatelské jméno a heslo (heslo je třeba napsat dvakrát, aby nedošlo k překlepům).

Účet administrátora bude vytvořen.



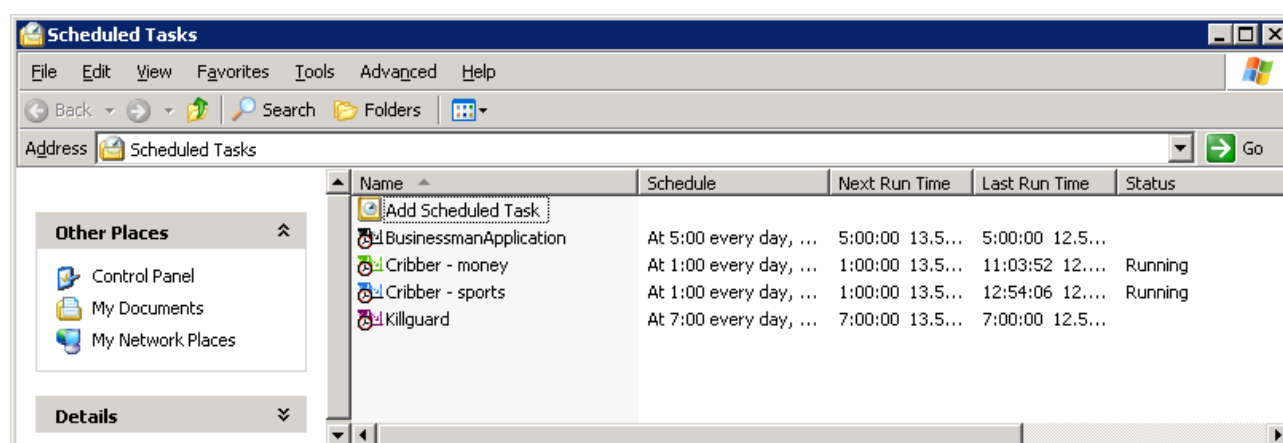
Obrázek 56

Tímto automatizovaná instalace končí. Je však třeba učinit ještě několik dalších kroků pro správné fungování systému.

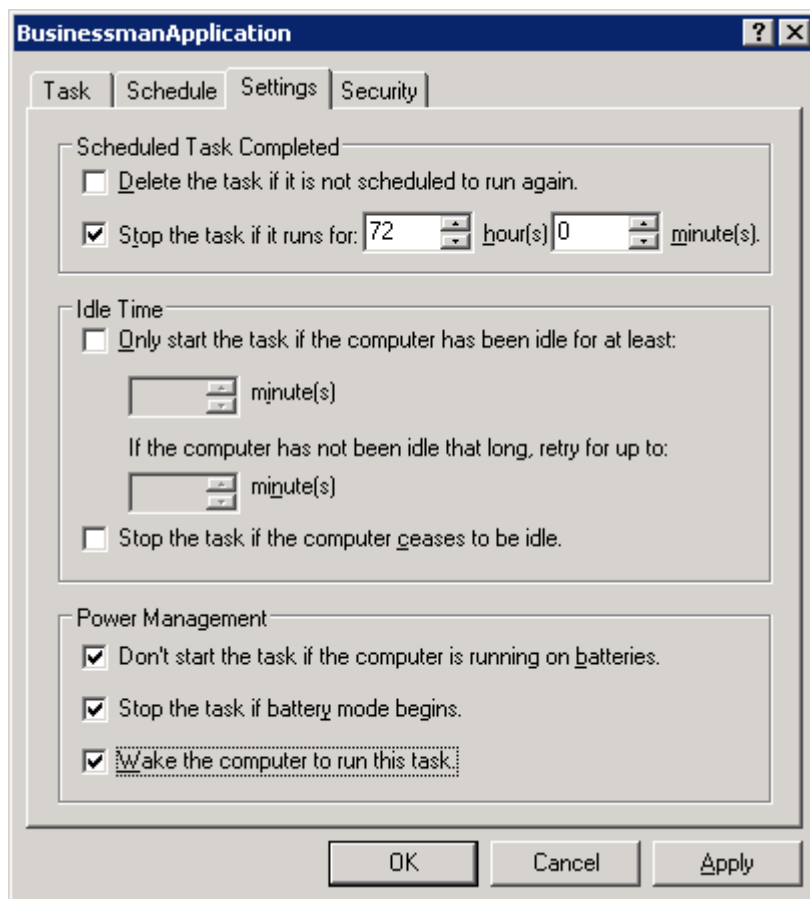


Obrázek 57

Pokud Váš server neuspáváte a běží 24 hodin denně, je možné následující krok vynechat. Pokud je server občas v úsporném režimu, je nutné ho na pravidelně spouštěné úlohy budit. Klikněte na *Start → All Programs → Accessories → System Tools → Scheduled Tasks*. Postupně projděte všechny čtyři úlohy *MoneyMakeru* (*BusinessmanApplication*, *Cribber – money*, *Cribber – sports* a *Killguard*), klikněte na ně pravým tlačítkem a zvolte *Properties*, nastavte v záložce *Settings* volbu *Wake the computer to run this task* (Obrázek 59).



Obrázek 58



Obrázek 59

Dále je třeba upravit nastavení *Apache* serveru*. V souboru `httpd.conf` nastavte cestu a práva do nainstalovaného adresáře `occupant`. Text 24 ukazuje správné nastavení.

Text 24: část `httpd.conf` souboru, změněné řádky jsou zvýrazněny

```
#
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "C:/MoneyMaker/occupant"
#
# Each directory to which Apache has access can be configured with respect
# to which services and features are allowed and/or disabled in that
# directory (and its subdirectories).
#
# First, we configure the "default" to be a very restrictive set of
# features.
#
<Directory />
    Options FollowSymLinks
    AllowOverride None
    Order deny,allow
    Allow from all
</Directory>
```

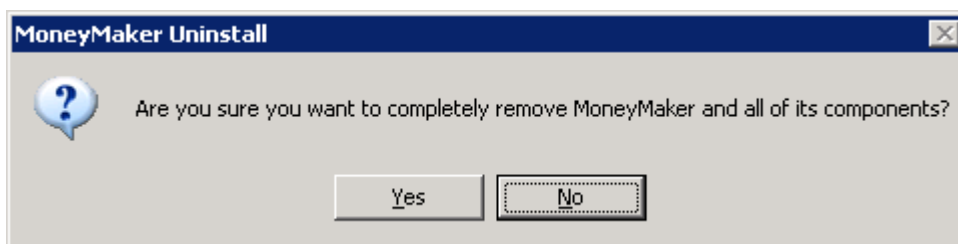
* Místo *Apache* serveru lze použít i *IIS* a obdobně ho nastavit.

Zbývá poslední krok – stáhnout na server dostatek dat. *Cribbery* se sice pravidelně spouštějí, ale stahují data za poslední měsíc, nebo rok, což je málo, pokud v databázi zatím žádná data nejsou. Je tedy třeba dočasně upravit nastavení *Cribberů* a ručně je spustit:

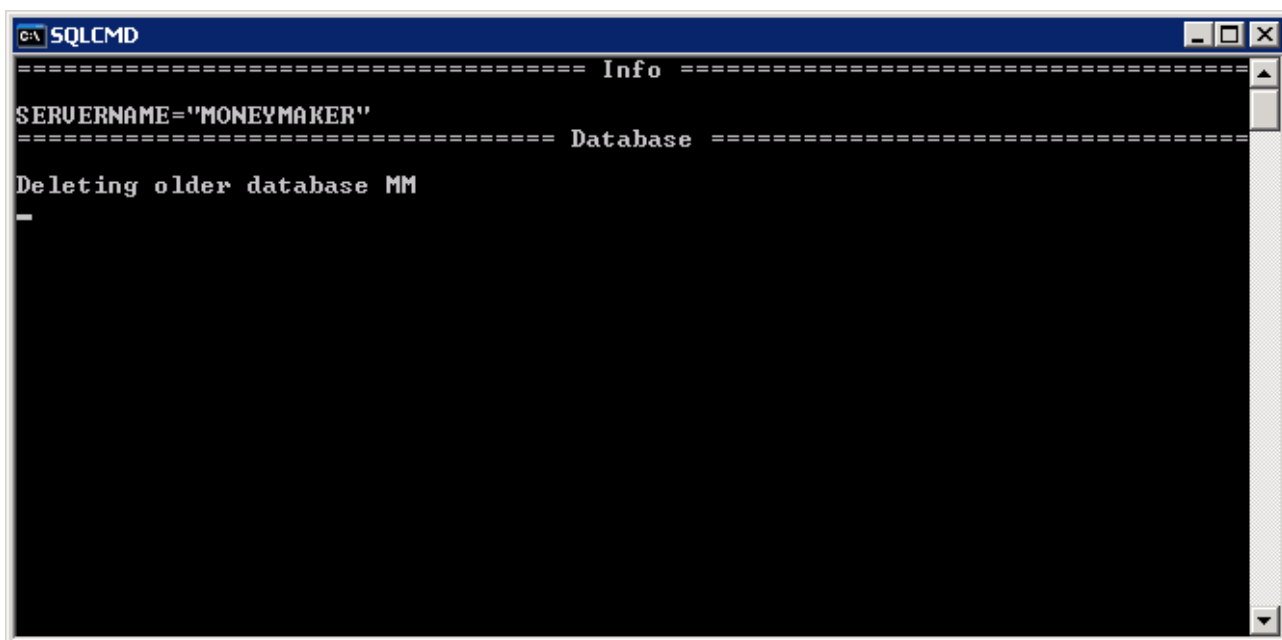
1. Otevřete si pro editaci konfigurační soubor C:\MoneyMaker\cribbers\cribber - money\MoneyDownload.exe.config.
2. Upravte hodnotu proměnné HistoryYears na hodnotu y, kde y=<letošní rok> - 1991.
3. Ručně spusťte *Cribber*, tedy spusťte program C:\MoneyMaker\cribbers\cribber - money\MoneyDownload.exe.
4. Poté, co *Cribber* skončí, vraťte konfigurační soubor po původního stavu, nastavte hodnotu proměnné HistoryYears např. na jedničku (jeden rok zpět).
5. Otevřete si pro editaci konfigurační soubor C:\MoneyMaker\cribbers\cribber - sports\SportsDownload.exe.config.
6. Upravte hodnotu proměnné HistoryDays na hodnotu např. 500.
7. Ručně spusťte *Cribber*, tedy spusťte program C:\MoneyMaker\cribbers\cribber - sports\SportsDownload.exe.
8. Poté, co *Cribber* skončí, vraťte konfigurační soubor po původního stavu, nastavte hodnotu proměnné HistoryDays např. na 30.

Deinstalace

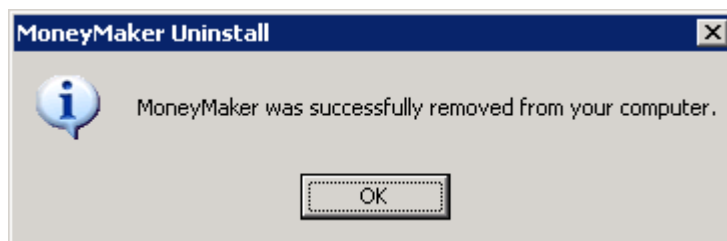
Pokud chcete *MoneyMaker* odinstalovat, použijte program Uninstall. Budete dotázáni, zda jste si jisti, že chcete systém odinstalovat. Pokud odpovíte kladně, systém bude odinstalován. Po ukončení práce se objeví informační dialog.



Obrázek 60



Obrázek 61



Obrázek 62

Odinstalační program smaže pouze komponenty, které byly nainstalované. Pokud si přejete smazat i všechny logy, instance strategií a podobně, je nutné to udělat ručně. Nejjednodušší způsob je smazat celý adresář, do kterého jste *MoneyMaker* nainstalovali.

Údržba systému

Systém není příliš náročný na údržbu. Především je vhodné pravidelně procházet logy a komprimovat je do archivu. Komprimování lze provádět pravidelně automaticky. Pročítání logů není zcela nezbytné – pokud nastane závažná chyba, bude administrátor informován emailem. Emaily jsou nastaveny v jednotlivých konfiguračních souborech a jednoduše je lze nastavit při instalaci.

Také je nutné ručně povolovat aktivitu nových uživatelů – více se dočtete v kapitole Bezpečnost (strana 116) a také v uživatelském návodu, v kapitole *Ovládání Moderátora pro administrátora nebo privilegovaného uživatele*.

Umístění logů

Zde uvedené cesty platí, pokud jste *MoneyMaker* nainstalovali do implicitně nabízené cesty C:\MoneyMaker\. Pokud jste zvolili jiné umístění, cesty se budou lišit.

<i>Businessman</i>	C:\MoneyMaker\Businessman\logs
<i>MoneyDownload</i>	C:\MoneyMaker\Cribber - money\logs
<i>SportDownload</i>	C:\MoneyMaker\Cribber - sports\logs
<i>Genesis</i>	C:\MoneyMaker\Genesis\logs
<i>Killguard</i>	C:\MoneyMaker\Killguard\logs

Konfigurační soubory

Zde uvedené cesty platí, pokud jste *MoneyMaker* nainstalovali do implicitně nabízené cesty C:\MoneyMaker\. Pokud jste zvolili jiné umístění, cesty se budou lišit.

<i>Businessman</i>	C:\MoneyMaker\Businessman\BusinessmanApplication.exe.config
<i>MoneyDownload</i>	C:\MoneyMaker\Cribber - money\MoneyDownload.exe.config
<i>SportDownload</i>	C:\MoneyMaker\Cribber - sports\SportsDownload.exe.config
<i>Genesis</i>	C:\MoneyMaker\Genesis\Genesis.exe.config
<i>Killguard</i>	C:\MoneyMaker\Killguard\killguard.exe.config C:\MoneyMaker\Killguard\killguard.vshost.exe.config

Co dělat když...

Nestačí počet připojení k databázi

V logu Businessmana se objeví následující řádek (administrátor je rovněž automaticky upozorněn emailem):

Text 25: řádek v logu obsahující chybu – důležitá část je zvýrazněna červenou barvou

```
Strategy 37: System.InvalidOperationException: Timeout expired. The timeout period
elapsed prior to obtaining a connection from the pool. This may have occurred because all
pooled connections were in use and max pool size was reached.
    at System.Data.ProviderBase.DbConnectionFactory.GetConnection(DbConnection
owningConnection)
    at System.Data.ProviderBase.DbConnectionClosed.OpenConnection(DbConnection
outerConnection, DbConnectionFactory connectionFactory)
```

```
at System.Data.SqlClient.SqlConnection.Open()  
at BusinessmanApplication.Program.Strategy.runAllInstances()
```

Znamená to, že byla přetížena databáze – bylo po ní požadováno více spojení, než je povoleno. Strategie běží paralelně a proto mohou spojení vyčerpat. Řešení jsou dvě:

1. snížit počet paralelních strategií, které mohou běžet (změna hodnoty proměnné `THREADSCOUNT` pomocí *Moderátora*)
2. zvýšit počet povolených otevřených připojení k databázi (v nastavení databáze)

Bezpečnost

Strategie uživatelů se nahrávají do systému jako knihovny a spouštějí se paralelně. Vystává otázka, zda může jedna strategie (knihovna) poškodit nebo ovlivnit jinou nebo dokonce celý systém.

Odpověď zní ano, ale pouze úmyslně. V současné době je bezpečnost implementována nastavením práv pomocí *.NET Access security code*. Chování knihovny je tedy omezené, ovšem není zakázáno, aby si sama knihovna práva změnila a umožnila si tak přístup k datům, ke kterým by přistupovat neměla.

Uživatel se nemusí bát, že by neúmyslně (chybou v pluginu) poškodil data ostatních uživatelů. Pokud by ale některý z uživatelů měl černé úmysly a chtěl data ostatních poškodit záměrně, v podstatě mu v tom nic nebrání. Proto je velmi důležité, aby administrátor povolil přístup do systému pouze důvěryhodným uživatelům.

Otázka bezpečnosti je na seznamu věcí, které by se měly do příští verze ještě lépe promyslet a implementovat. V současné verzi je popisované omezení práv implementováno v souboru `PluginBase.cs`.