Návrhová dokumentace

***Projekt volební systém***

*Vypracoval Petr Hlaváček, Ondřej Kulatý, Štěpán Škorpil a Václav Tarantík*

**Obsah:**

Balíček Class Model 3

Balíček core 3

SystemRegException 3

Balíček data\_tier 4

ActionDAO 4

ParticipantDAO 7

UserDAO 10

Balíček Entities 11

Action 12

Participant 14

Presence 16

User 17

Balíček Enums 18

Role 18

Balíček business\_tier 19

ActionFacade 20

ParticipantFacade 23

UserFacade 28

Balíček Utils 29

CalendarUtil 29

Generator 30

## Balíček Class Model

Balíček s modely balíčků a tříd, které se nacházejí v aplikaci registrační systém. V systému se zatím nacházejí 2 vrstvy a to datová v balíčku data\_tier a business logika v balíčku business\_tier.

**Diagram: Class Model**

Created By: Lahvion 19.11.2005

Last Modified: 2.11.2011

Popis:

Model všech balíčků a jejich tříd dohromady.



Figure: Diagram Class Model

#### Balíček core

V balíčku core, se nacházejí nejzákladnější třídy pro práci se systémem na nejnižší úrovni. Je v něm balíček data\_tier s DAO třídami a entitami a pak je zde třída reprezentující systémovou výjimku.

**Diagram: core**

Created By: Lahvion 2.11.2011

Last Modified: 2.11.2011

Popis:



Figure: Diagram core

#### *SystemRegException*

Popis:

Třída zděděná od java.lang.Exception, reprezentuje všechny výjimky, které mohou být v systému zachyceny.

#### Balíček data\_tier

V balíčku data\_tier jsou Data Access Object třídy, které přímo pracují s databází, či jiným úložištěm. Pro každou entitu z balíčku Entities je zde jedna DAO třída, která s entitou pracuje.

V každé DAO třídě se vyskytují metody jako create get change a delete, které pracují s danou entitou, k níž se třída vztahuje. Například v třídě ActionDAO je metoda createAction, která vytvoří novou entitu akce a uloží ji do databáze.

**Diagram: data\_tier**

Created By: Lahvion 2.11.2011

Last Modified: 2.11.2011

Popis:



Figure: Diagram data\_tier

##### ActionDAO

Popis:

Třída určená pro datovou práci s entitními objekty typu Action.

**Atribut: dbConnection** Private

Typ: Connection

Popis: Připojení k databázi, do které se budou akce ukládat.

**Atribut: actions** Private

Typ: Map<Integer, Action>

Popis: Mapa akcí. Klíčem bude identifikátor akce a hodnota bude instance akce. Může se používat pro rychlejší práci, aby se nemusel sahat do databáze. Při vytváření instance třídy se do mapy načtou akce s databáze

**Metoda: createAction** Public

Popis: Vytvoří novu volební akci. V parametrech obdrží název akce, místo konání, datum zahájení datum ukončení a stručný popis akce. Popis akce není povinný atribut a může být null.

***Parametr: name***

Typ: String

Popis: Název nové akce.

***Parametr: place***

Typ: String

Popis: Místo konání nové akce.

***Parametr: startDate***

Typ: Calendar

Popis: Datum a čas zahájení nové akce.

***Parametr: endDate***

Typ: Calendar

Popis: Datum a čas ukončení nové akce.

***Parametr: description***

Typ: String

Popis: Stručný popis dané akce.

**Metoda: deleteAction** Public

Popis: Metoda vyhledá v databázi akci podle jejího jednoznačného id a následně ji odstraní.

***Parametr: id***

Typ: int

Popis: Identifikátor odstraňované akce.

**Metoda: getAction** Public

Popis: Vyhledá a vrátí akci s id získaným z parametru id.

***Parametr: id***

Typ: int

Popis: Identifikátor vyhledávané akce.

**Metoda: getAction** Public

Popis: Metoda vyhledá a vrátí akci, která má název jako parametr name.

***Parametr: name***

Typ: String

Popis: Název vyhledávané akce.

**Metoda: setDates** Public

Popis: Metoda nastavuje nové data a časy pro konec a start akce. Parametru startDate, nastavuje novou dobu zahájení a endDate, který nastavuje novou dobu ukončení akce. Pokud chceme měnit pouze jeden z těchto dvou parametrů tak ten druhý nastavíme na null. Akci vyhledáme podle parametru id.

***Parametr: id***

Typ: int

Popis: Id akce, které chceme měnit data.

***Parametr: startDate***

Typ: Calendar

Popis: Nové datum a čas zahájení akce, pokud nechceme měnit, předáme null.

***Parametr: endDate***

Typ: Calendar

Popis: Nové datum a čas ukončení akce, pokud nechceme měnit, předáme null.

**Metoda: setParametrs** Public

Popis: Metoda edituje parametry akce, která má id shodné s parametrem id. Nastavuje název, místo a popis akce. Pokud některý atribut nechceme měnit, necháme ho null.

***Parametr: desc***

Typ: String

Popis: Nový popis akce.

***Parametr: place***

Typ: String

Popis: Nové místo konání akce.

***Parametr: name***

Typ: String

Popis: Nový název akce,

***Parametr: id***

Typ: int

Popis: Identifikátor editované akce.

**Metoda: addParticipants** Public

Popis: Metoda přidává účastníka s přihlašovacím jménem login do akce s identifikátorem id.

***Parametr: id***

Typ: int

Popis: Identifikátor akce, které chceme přidat účastníka.

***Parametr: login***

Typ: String

Popis: Login účastníka, který bude přidán do seznamu účastníků akce.

**Metoda: deleteParticipant** Public

Popis: Odstraní login účastníka, o kterém chceme říci, že se akce nebude účastnit.

***Parametr: participant***

Typ: String

Popis: Login odstraňovaného účastníka.

##### ParticipantDAO

Popis:

Třída určená pro datovou práci s entitními objekty typu Participants.

**Atribut: participants** Private

Typ: Map<Integer, Participant>

Popis: Mapa účastníka, kde klíč je id účastníka a hodnota je instance daného účastníka. Při vytváření instance třídy ParticipantDAO se do mapy načtou účastníci z databáze.

**Atribut: dbConnection** Private

Typ: Connection

Popis: Databázové spojení.

**Metoda: createParticipant** Public

Popis: Vytvoří nového účastníka podle parametrů id, name a string. Login a heslo se uživateli vygenerují až při dokončení registrace.

***Parametr: address***

Typ: String

Popis: Adresa nového účastníka.

***Parametr: id***

Typ: int

Popis: Jednoznačný identifikátor (číslo OP) nového účastníka.

***Parametr: name***

Typ: String

Popis: Jméno nového účastníka.

**Metoda: getParticipant** Public

Popis: Vrací účastníka s identifikátorem předaném v parametru id.

***Parametr: id***

Typ: int

Popis: Id účastníka.

**Metoda: getParticipant** Public

Popis: Vyhledá a vrátí uživatele s přihlašovacím jménem z parametru login.

***Parametr: login***

Typ: String

Popis: Login hledaného účastníka.

**Metoda: deleteParticipant** Public

Popis: Vyhledá účastníka podle id z parametru a poté ho odstraní.

***Parametr: id***

Typ: int

Popis: Identifikátor mazaného účastníka.

**Metoda: deleteParticipant** Public

Popis: Vyhledá účastníka podle přihlašovacího jména z parametru login a poté ho odstraní.

***Parametr: login***

Typ: String

Popis: Login odstraňovaného účastníka.

**Metoda: changeLogin** Public

Popis: Metoda vyhledá účastníka podle id v parametru id a změní mu heslo a login podle parametrů login a password Pokud chceme měnit jen jeden z těchto atributů, tak ten druhý necháme null.

***Parametr: login***

Typ: String

Popis: Nový login daného účastníka.

***Parametr: password***

Typ: String

Popis: Nové heslo daného účastníka.

***Parametr: id***

Typ: int

Popis: Číslo OP účastníka, kterému chceme měnit přihlašovací údaje.

**Metoda: changeParams** Public

Popis: Metoda vyhledá účastníka podle id v parametru id a změní mu adresu a jméno podle parametrů name a address, pokud chceme nastavovat jen jeden z těchto parametrů, tak ten druhý nastavíme na null.

***Parametr: address***

Typ: String

Popis: Nová adresa účastníka.

***Parametr: name***

Typ: String

Popis: Nové jméno účastníka.

***Parametr: int***

Typ: id

Popis: Číslo OP účastníka, kterému údaje měníme.

**Metoda: changeParams** Public

Popis: Stejná metoda jako metoda changeParams(String, String, int), ale daný účastník se bude vyhledávat podle jeho loginu.

***Parametr: address***

Typ: String

Popis: Nová adresa.

***Parametr: name***

Typ: String

Popis: Nové jméno.

***Parametr: login***

Typ: String

Popis: Login, podle kterého se účastník vyhledá.

**Metoda: changeLogin** Public

Popis: Stejná metoda jako changeLogin(String, String, int), ale daný účastník se bude hledat podle jeho starého loginu oldLogin.

***Parametr: login***

Typ: String

Popis: Nový login

***Parametr: password***

Typ: String

Popis: Nové heslo.

***Parametr: oldLogin***

Typ: String

Popis: Starý login, podle kterého se účastník vyhledá.

**Metoda: getPresence** Public

Popis: Metoda vyhledá účastníka podle jeho id a vrátí objekt s jeho účastí.

***Parametr: id***

Typ: int

Popis: Číslo OP vyhledávaného účastníka.

**Metoda: getPresence** Public

Popis: Metoda vyhledá účastníka podle jeho loginu a vrátí objekt s jeho účastí.

***Parametr: login***

Typ: String

Popis: Login vyhledávaného účastníka.

##### UserDAO

Popis:

Třída určená pro datovou práci s entitními objekty typu User.

**Atribut: dbConnection** Private

Typ: Connection

Popis: Připojení k databázi, do níž se budou uživatelé ukládat.

**Atribut: users** Private

Typ: Map<String, User>

Popis: Mapa uživatelů, kde klíč je login uživatele a hodnota je instance daného uživatele. Při vytváření instance třídy UserDAO se do mapy načtou uživatelé z databáze.

**Metoda: createUser** Public

Popis: Vytvoří nového uživatele. Uživatel bude mít roli podle atributu role, login podle login a heslo podle password.

***Parametr: role***

Typ: Role

Popis:

***Parametr: login***

Typ: String

Popis:

***Parametr: password***

Typ: String

Popis:

**Metoda: deleteUser** Public

Popis: Vyhledá uživatele podle loginu, předaném parametrem, a odstraní ho z databáze.

***Parametr: login***

Typ: String

Popis:

**Metoda: getUser** Public

Popis: Vyhledá uživatele podle parametru login a vrátí ho.

***Parametr: login***

Typ: String

Popis: Login požadovaného uživatele.

**Metoda: changeLogin** Public

Popis: Vyhledá uživatele podle atributu oldLogin a nastaví mu nové přístupové údaje login a password. Pokud chceme měnit jen jeden z nich, tak druhý parametr musí být null.

***Parametr: login***

Typ: String

Popis: Login uživatele, kterému chceme změnit heslo.

***Parametr: password***

Typ: String

Popis: Nové heslo uživatele

***Parametr: oldLogin***

Typ: String

Popis: Login účastníka, kterému měníme přístupové údaje.

**Metoda: changeRole** Public

Popis: Vyhledá uživatele podle loginu v atributu login a nastaví mu novou roli podle atributu role.

***Parametr: login***

Typ: String

Popis: Login uživatele, kterému chceme nastavovat novou roli.

***Parametr: role***

Typ: Role

Popis: Nová role uživatele.

##### Balíček Entities

V tomto balíčku se nachází základní entity, které se v aplikaci registrační systém vyskytují. Dále je zde ještě balíček enums, kde se nacházejí výčty potřebné pro entity.

**Diagram: Entities**

Created By: Lahvion 2.11.2011

Last Modified: 2.11.2011

Popis:



Figure: Diagram Entities

###### *Action*

Popis:

Entita akce jednoduše reprezentuje volební akci. Je svázána s entitou účastník relací 0 ..\* ku 1 .. \*, to znamená že nově vytvořená akce nemusí mít nutně přiřazené účastníky a poté se jich akci může přiřadit libovolný počet.

**Atribut: description** Private

Typ: String

Popis: Stručný popis akce.

**Atribut: endDate** Private

Typ: Calendar

Popis: Datum a čas ukončení akce.

**Atribut: participants** Private

Typ: List<String>

Popis: Seznam loginů účastníků, kteří se na akci přihlásili.

**Atribut: place** Private

Typ: String

Popis: Místo konání akce.

**Atribut: startDate** Private

Typ: Calendar

Popis: Datum a čas zahájení akce.

**Atribut: name** Private

Typ: String

Popis: Název akce.

**Metoda: getDescription** Public

Popis: Vrací stručný popis akce.

**Metoda: getEndDate** Public

Popis: Vrací datum a čas ukončení akce.

**Metoda: getStartDate** Public

Popis: Vrací datum a čas zahájení akce.

**Metoda: getParticpants** Public

Popis: Vrací seznam loginů účastníků dané akce.

**Metoda: getPlace** Public

Popis: Vrací místo konání dané akce.

**Metoda: getName** Public

Popis: Vrací název akce.

**Metoda: setDescription** Public

Popis: Nastavuje nový popis akce, který se získá z parametru desc.

***Parametr: desc***

Typ: String

Popis: Nový popis akce.

**Metoda: setEndDate** Public

Popis: Nastavuje nový datum a čas ukončení dané akce. Nové hodnoty jsou předány parametrem endDate.

***Parametr: endDate***

Typ: Calendar

Popis: Nový datum a čas ukončení akce

**Metoda: setStartDate** Public

Popis: Nastavuje nový datum a čas zahájení dané akce. Nové hodnotu se získají parametrem startDate.

***Parametr: startDate***

Typ: Calendar

Popis: Nový datum a čas zahájení akce.

**Metoda: setPlace** Public

Popis: Metoda nastavuje nové místo konání akce, Nová hodnota je předána parametrem place.

***Parametr: place***

Typ: String

Popis: Nové místo konání akce.

**Metoda: setName** Public

Popis: Metoda nastavuje dané akci nové jméno, které obdrží v parametru name.

***Parametr: name***

Typ: String

Popis: Nové jméno akce.

**Metoda: addParticipants** Public

Popis: Přidává login účastníka, který se bude akce účastnit.

***Parametr: login***

Typ: String

Popis: Login přidávaného účastníka.

###### *Participant*

Popis:

Entita účastník reprezentuje člověka, jenž se bude účastnit dané akce. Je asociována s entitami akce a s entitou přítomnost. Každý uživatel musí být přiřazen nějaké akci a každý uživatel má právě jednu přítomnost.

Účastník má v sobě základní údaje o osobě jako Jméno, Adresa, číslo občanského průkazu a případně další libovolné akce. Dále pak heslo a id, které se uživateli vygenerují, když se dostaví na akci, kde je zaregistrován.

**Atribut: id** Private

Typ: String

Popis:

**Atribut: name** Private

Typ: String

Popis: Jméno účastníka.

**Atribut: address** Private

Typ: String

Popis: Adresa účastníka.

**Atribut: Login** Private

Typ: String

Popis: Přihlašovací jméno účastníka.

**Atribut: password** Private

Typ: String

Popis: Heslo účastníka.

**Atribut: presence** Private

Typ: Presence

Popis: Objekt reprezentující presenci uživatele.

**Atribut: completeReg** Private

Typ: boolean

Popis: Příznak zda byla registrace uživatele již dokončena nebo ještě ne.

**Metoda: getPersonalID** Public

Popis: Vrací číslo OP daného účastníka.

**Metoda: getName** Public

Popis: Vrací jméno daného účastníka.

**Metoda: getAddress** Public

Popis: Vrací adresu účastníka.

**Metoda: getLogin** Public

Popis: Vrací přihlašovací jméno účastníka.

**Metoda: getPassword** Public

Popis: Vrací heslo účastníka.

**Metoda: getPresence** Public

Popis: Vrací objekt reprezentující presenci účastníka.

**Metoda: isCompleteReg** Public

Popis: Vrací true pokud má uživatel již dokončenou registraci, false pokud ne.

**Metoda: setName** Public

Popis: Metoda nastavuje uživateli nové jméno, které získá z parametru name.

***Parametr: name***

Typ: String

Popis: Nové jméno účastníka.

**Metoda: setAddress** Public

Popis: Metoda nastavuje novou adresu účastníka, kterou získá z atributu address.

***Parametr: address***

Typ: String

Popis: Nová adresa účastníka.

**Metoda: setLogin** Public

Popis: Metoda nastavuje účastníkovi nové uživatelské jméno, které získá z atributu login.

***Parametr: login***

Typ: String

Popis: Nový login účastníka.

**Metoda: setPassword** Public

Popis: Metoda nastavuje účastníkovi nové heslo, které získá z parametru password.

***Parametr: password***

Typ: String

Popis: Nové heslo.

**Metoda: setCompleteReg** Public

Popis: Metoda nastavuje příznak completeReg na true, čímž říká, že daný účastník má kompletní registaci.

###### *Presence*

Popis:

Entita přítomnost je svázána s entitou účastníky relací One to One. Každý účastník tedy má právě jednu přítomnost a každá přítomnost náleží právě jednomu účastníkovi.

**Atribut: odchody** Private

Typ: List<Calendar>

Popis: Seznam odchodů daného účastníka. Při každém odchodu se sem zaznamená čas a datum odchodu.

**Atribut: prichody** Private

Typ: List<Calendar>

Popis: Seznam s příchody daného účastníka. Při každém příchodu na akci, se do něj účastníkovi zaznamená datum a čas příchodu.

**Atribut: presence** Private

Typ: boolean

Popis: Booleanovská proměnná, která udává, zda je uživatel na akci přítomen právě teď či nikoliv. Pokud je přítomen je proměnná true, jinak je false.

**Metoda: isPresent** Public

Popis: Vrací true nebo false, podle toho jestli je uživatel na akci přítomen právě teď.

**Metoda: setPresence** Public

Popis: Podle parametru presence se nastaví přítomnost uživatele. Používá se, pokud zrovna uživatel přišel nebo odešel.

***Parametr: presence***

Typ: boolean

Popis: Nová přítomnost účastníka. Pokud je true, tak právě přišel, pokud false tak odešel.

**Metoda: getArrivals** Public

Popis: Vrací seznam příchodů daného uživatele.

**Metoda: getDepartures** Public

Popis: Vrací seznam odchodů.

###### *User*

Popis:

Administrátor je entita, která reprezentuje osobu, jež se systémem pracuje. Jelikož jsou v systému tři aktéři, entita mezi nimi rozlišuje podle proměnné Role role, která má tři hodnoty: SUPER\_ADMIN, ADMIN a REGISTRATOR.

Dále už je pak v entitě pouze heslo a login, tedy údaje, kterými se budou uživatelé do systému přihlašovat.

**Atribut: login** Private

Typ: String

Popis:

**Atribut: password** Private

Typ: String

Popis: Heslo účastníka.

**Atribut: role** Private

Typ: Role

Popis: Atribut, který reprezentuje roli uživatele v systému.

**Metoda: getLogin** Public

Popis: Vrací login usera.

**Metoda: getPassword** Public

Popis: Vrací heslo usera.

**Metoda: getRole** Public

Popis: Vrací uživatelskou roli daného uživatele.

**Metoda: setRole** Public

Popis: Nastaví danému uživateli novou roli.

***Parametr: role***

Typ: Role

Popis: Nová role uživatele.

**Metoda: setPassword** Public

Popis: Nastaví danému uživateli nové heslo.

***Parametr: password***

Typ: String

Popis: Nové heslo.

**Metoda: setLogin** Public

Popis: Nastaví nový login daného usera.

***Parametr: login***

Typ: String

Popis: Nové přihlašovací jméno daného uživatele.

###### Balíček Enums

V balíčku je výčet Role, který používá entita User pro identifikování uživatelské role.

**Diagram: Enums**

Created By: Lahvion 2.11.2011

Last Modified: 2.11.2011

Popis:



Figure: Diagram Enums

*Role*

«enumeration»

Popis:

Výčet Role má tři hodnoty, které reprezentují role v registračním systému.

Pro roli Registrátor má výčet hodnotu REGISTRATOR.

Pro roli Administrátor má výčet hodnotu ADMIN.

Pro roli SuperAdmin má výčet hodnotu SUPER\_ADMIN.

**Atribut: SUPER\_ADMIN** Public

Typ:

Popis: Hodnota pro roli reprezentující SuperAdmina.

**Atribut: ADMIN** Public

Typ:

Popis: Hodnota pro roli reprezentující Admina.

**Atribut: REGISTRATOR** Public

Typ:

Popis: Hodnota pro roli reprezentující Registrátora.

#### Balíček business\_tier

V balíčku business\_tier bude implementována business logika aplikace. Je to jakási fasáda pro práci s nižší, datovou, vrstvou.

**Diagram: business\_tier**

Created By: Lahvion 2.11.2011

Last Modified: 2.11.2011

Popis:



Figure: Diagram business\_tier

#### *ActionFacade*

Popis:

Třída poskytuje business logiku pro práci s entitou akce Action.

**Atribut: actionDataAccess** Private

Typ: ActionDAO

Popis: Instance data access objektu pro entitu Action, která bude business logice v třídě ActionFacade poskytovat metody pro práci s databází.

**Metoda: createAction** Public

Popis: Metoda požádá třídu ActionDAO o vytvoření nové akce bez popisu.

***Parametr: name***

Typ: String

Popis: Název akce.

***Parametr: place***

Typ: String

Popis: Místo konání akce.

***Parametr: endDate***

Typ: Calendar

Popis: Datum a čas ukončení akce,

***Parametr: startDate***

Typ: Calendar

Popis: Datum a čas zahájení akce,

**Metoda: createAction** Public

Popis: Metoda požádá třídu ActionDAO o vytvoření nové akce se všemi parametry.

***Parametr: name***

Typ: String

Popis: Název akce.

***Parametr: place***

Typ: String

Popis: Místo konání akce.

***Parametr: startDate***

Typ: Calendar

Popis: Datum a čas zahájení akce.

***Parametr: endDate***

Typ: Calendar

Popis: Datum a čas ukončení akce,

***Parametr: desc***

Typ: String

Popis: Stručný popis akce.

**Metoda: deleteAction** Public

Popis: Metoda požádá třídu ActionDAO o odstranění akce s identifikátorem id předaným parametrem.

***Parametr: id***

Typ: int

Popis: Identifikátor odstraňované akce.

**Metoda: findAction** Public

Popis: Metoda požádá třídu ActionDAO o vyhledání akce s identifikátorem z parametru id.

***Parametr: id***

Typ: int

Popis: Identifikátor hledané akce.

**Metoda: findAction** Public

Popis: Metoda požádá třídu ActionDAO o vyhledání akce s názvem z parametru name.

***Parametr: name***

Typ: String

Popis: Název hledané akce.

**Metoda: changePlace** Public

Popis: Požádá třídu ActionDAO o změnu místa konání akce s id z parametru id.

***Parametr: id***

Typ: int

Popis: ID editované akce.

***Parametr: place***

Typ: String

Popis: Nové místo editované akce.

**Metoda: changeName** Public

Popis: Požádá třídu ActionDAO o změna názvu akce, která se vyhledá podle parametru ID.

***Parametr: name***

Typ: String

Popis: Nový název akce.

***Parametr: ID***

Typ: int

Popis: ID editované akce.

**Metoda: changeDesc** Public

Popis: Požádá třídu ActionDAO o změnu stručného popisu akce, která se vyhledá podle identifikátoru z parametru ID.

***Parametr: ID***

Typ: int

Popis: ID editované akce.

***Parametr: desc***

Typ: String

Popis: Nový popis akce.

**Metoda: changeParams** Public

Popis: Kombinuje předchozí change metody dohromady. String parametry mohou být null a pak se nebudou měnit.

***Parametr: ID***

Typ: int

Popis: ID editované akce.

***Parametr: name***

Typ: String

Popis: Nové jméno akce.

***Parametr: place***

Typ: String

Popis: Nové místo konání akce.

***Parametr: desc***

Typ: String

Popis: Nový popisek akce.

**Metoda: getParticipants** Public

Popis: Požádá třídu ActionDAO o vyhledání a vrácení loginů všech účastníků, kteří se akce vyhledané podle ID z parametru id akce účastní.

***Parametr: id***

Typ: int

Popis: ID požadované akce.

**Metoda: addParticipant** Public

Popis: Požádá třídu ActionDAO o přidání uživatele s loginem získaným z parametru login na akci vyhledanou podle parametru id.

***Parametr: id***

Typ: int

Popis: ID dané akce.

***Parametr: login***

Typ: String

Popis: Login uživatele, kterého na akci přidáváme.

**Metoda: removeParticipant** Public

Popis: Požádá třídu ActionDAO o odstranění účastníka s loginem login z akce, která má ID id.

***Parametr: login***

Typ: String

Popis: Login odstraňovaného účastníka.

***Parametr: id***

Typ: int

Popis: ID dané akce.

#### *ParticipantFacade*

Popis:

Třída poskytující business logiku pro práci s entitou účastníka Participant.

**Atribut: participantDataAccess** Private

Typ: ParticipantDAO

Popis: Instance data access objektu pro entitu Participant, která bude business logice ve třídě ParticipantFacade poskytovat metody pro práci s databází.

**Metoda: changePassword** Public

Popis: Metoda požádá třídu ParticipantDAO o změnu hesla účastníka s loginem v parametru login.

***Parametr: password***

Typ: String

Popis: Nové heslo.

***Parametr: login***

Typ: String

Popis: Jméno editovaného účastníka.

**Metoda: changeLogin** Public

Popis: Metoda požádá třídu ParticipantDAO o změnu loginu akce s loginem v parametru oldLogin.

***Parametr: login***

Typ: String

Popis: Nový login účastníka.

***Parametr: oldLogin***

Typ: String

Popis: Starý login editovaného účastníka.

**Metoda: changeLogin** Public

Popis: Metoda požádá třídu PaticipantDAO o změnu přihlašovacích údajů účastníka vyhledaného podle atributu oldLogin. Pokud chceme měnit pouze jeden z atributů login a password, tak ten nežádoucí musí být null.

***Parametr: login***

Typ: String

Popis: Nový login účastníka.

***Parametr: password***

Typ: String

Popis: Nové heslo účastníka.

***Parametr: oldLogin***

Typ: String

Popis: Starý login editovaného účastníka.

**Metoda: createParticipant** Public

Popis: Metoda požádá třídu ParticipantDAO o vytvoření nového účastníka, se zadaným jménem, adresou a číslem OP.

***Parametr: name***

Typ: String

Popis: Jméno nového účastníka.

***Parametr: address***

Typ: String

Popis: Adresa nového účastníka.

***Parametr: personalID***

Typ: int

Popis: Číslo OP nového účastníka.

**Metoda: findParticipant** Public

Popis: Metoda požádá třídu ParticipantDAO o vyhledání a vrácení účastníka s číslem OP z parametru id.

***Parametr: id***

Typ: int

Popis: Číslo OP hledaného účastníka.

**Metoda: findParticipant** Public

Popis: Metoda požádá třídu ParticipantDAO o vyhledání a vrácení účastníka s loginem z parametru login.

***Parametr: login***

Typ: String

Popis: Login hledaného účastníka.

**Metoda: deleteParticipant** Public

Popis: Metoda požádá třídu ParticipantDAO o odstranění účastníka, kterého vyhledá podle čísla OP id.

***Parametr: id***

Typ: int

Popis: Číslo OP odstraňovaného účastníka.

**Metoda: deleteParticipant** Public

Popis: Metoda požádá třídu ParticipantDAO o odstranění účastníka vyhledaného podle parametru login.

***Parametr: login***

Typ: String

Popis: Login odstraňovaného účastníka.

**Metoda: completeReg** Public

Popis: Metoda požádá třídu ParticipantDAO o dokončení registrace účastníka vyhledaného podle parametru id. V metodě se vygeneruje heslo a login účastníka a požádá se o nastavení těchto parametrů a také o nastavení příznaku o dokončení registrace.

***Parametr: id***

Typ: int

Popis: Číslo OP daného účastníka.

**Metoda: changeAddress** Public

Popis: Metoda požádá třídu ParticipantDAO o změnu adresy účastníka vyhledaného parametru login.

***Parametr: login***

Typ: String

Popis: Login editovaného účastníka.

***Parametr: address***

Typ: String

Popis: Nová adresa.

**Metoda: changeName** Public

Popis: Metoda požádá třídu ParticipantDAO o změnu jména účastníka vyhledaného parametru login.

***Parametr: name***

Typ: String

Popis: Nové jméno.

***Parametr: login***

Typ: String

Popis: Login editovaného účastníka.

**Metoda: changeParams** Public

Popis: Metoda kombinuje předchozí dvě metody pro změnu adresy a jména. Opět se vyhledává podle loginu.

***Parametr: address***

Typ: String

Popis: Nová adresa.

***Parametr: name***

Typ: String

Popis: Nové jméno.

***Parametr: login***

Typ: String

Popis: Login editovaného účastníka.

**Metoda: changeAddress** Public

Popis: Stejné jako předchozí změna adresy, ale vyhledává se podle čísla OP.

***Parametr: id***

Typ: int

Popis:

**Metoda: changeName** Public

Popis: Stejné jako předchozí změna jména, ale vyhledává se podle čísla OP.

***Parametr: name***

Typ: String

Popis: Nové jméno.

***Parametr: id***

Typ: int

Popis: Číslo OP editovaného účastníka.

**Metoda: changeParams** Public

Popis: Stejné jako předchozí metody pro změnu adresy a jména ale zkombinované dohromady.

***Parametr: name***

Typ: String

Popis: Nové jméno.

***Parametr: address***

Typ: String

Popis: Nová adresa.

***Parametr: id***

Typ: int

Popis: Číslo OP editovaného účastníka.

**Metoda: isPresent** Public

Popis: Metoda vrací true nebo false, podle aktuální účasti účastníka, o jehož vyhledání podle parametru login je požádána třída ParticipantDAO, na akci.

***Parametr: login***

Typ: String

Popis: Login daného účastníka.

**Metoda: isPresent** Public

Popis: To samé jako předchozí verze, ale vyhledává se podle čísla OP.

***Parametr: id***

Typ: int

Popis: Číslo OP daného účastníka.

**Metoda: wasPresent** Public

Popis: Metoda požádá třídu ParticipantDAO o vrácení objektu s presencí daného uživatele, vyhledaného podle loginu login a poté pomocí metody třídy CalendarUtil vypočítá, zda byl v čase určeném parametrem time přítomen či nikoliv.

***Parametr: time***

Typ: Calendar

Popis: Datum a čas hledaného doby.

***Parametr: login***

Typ: String

Popis: Login daného účastníka.

**Metoda: wasPresent** Public

Popis: Stejné jako předchozí verze metody, ale vyhledává se podle čísla OP.

***Parametr: time***

Typ: Calendar

Popis: Doba hledané přítomnosti.

***Parametr: id***

Typ: int

Popis: Číslo OP daného účastníka.

#### *UserFacade*

Popis:

Fasáda pro práci s uživateli.

**Atribut: userDataAccess** Private

Typ: UserDAO

Popis: Instance data access objektu pro entitu User, která bude business logice poskytovat metody pro práci s databází.

**Metoda: createSuperAdmin** Public

Popis: Požádá UserDAO o vytvoření nového uživatele s rolí SuperAdmin.

**Metoda: createAdmin** Public

Popis: Požádá UserDAO o vytvoření nového uživatele s rolí Admin.

**Metoda: createRegistator** Public

Popis: Požádá UserDAO o vytvoření nového uživatele s rolí Registrator.

**Metoda: findUser** Public

Popis: Požádá UserDAO o vyhledání uživatele s loginem z parametru login.

***Parametr: login***

Typ: String

Popis:

**Metoda: changePassword** Public

Popis: Požádá UserDAO o změnu hesla uživatele s daným loginem login.

***Parametr: login***

Typ: String

Popis: Login uživatele.

***Parametr: password***

Typ: String

Popis: Nové heslo.

**Metoda: changeLogin** Public

Popis: Požádá UserDAO o změnu loginu daného uživatele s loginem oldLogin.

***Parametr: login***

Typ: String

Popis: Nové heslo.

***Parametr: oldLogin***

Typ: String

Popis: Staré heslo, podle kterého se uživatel vyhledá,

**Metoda: changeLogin** Public

Popis: Požádá UserDAO o změnu přihlašovacích údajů (heslo a login) uživatele s loginem z parametru login.

***Parametr: login***

Typ: String

Popis: Nový login.

***Parametr: password***

Typ: String

Popis: Nové heslo.

***Parametr: oldLogin***

Typ: String

Popis: Login uživatele, jehož přihlašovací údaje chceme měnit.

**Metoda: deleteUser** Public

Popis: Požádá třídu UserDAO o odstranění uživatele s loginem login.

***Parametr: login***

Typ: String

Popis: Login, podle kterého se odstraňovaný uživatel vyhledá.

#### Balíček Utils

**Diagram: Utils**

Created By: Lahvion 2.11.2011

Last Modified: 2.11.2011

Popis:



Figure: Diagram Utils

##### CalendarUtil

Popis:

Třída má metodu, které počítá s daty a umí vypočítat, zda byl v tu kterou dobu zákazník přítomen či nepřítomen.

**Metoda: wasPresent** Public Static

Popis: Metoda podle seznamu příchodů, odchodů a podle daného času pro daného účastníka zjistí, zda byl na akci přítomen nebo ne.

***Parametr: arrivals***

Typ: List<Calendar>

Popis: List s daty a časy příchodů.

***Parametr: departures***

Typ: List<Calendar>

Popis: List s daty a časy odchodů

***Parametr: currentTime***

Typ: Calendar

Popis: Datum a čas pro který chceme zjisti zda byl účastník přítomen či nepřítomen

##### Generator

Popis:

Třída pro generování náhodných řetězců. Její metody se budou používat při potřebě vygenerovat login nebo heslo nějakému účastníkovi či uživateli.

**Atribut: LENGTH** Protected Static

Typ: int

Popis: Implicitní délka generovaných řetězců. Je protected, lze tedy nastavit z nějaké business třídy.

**Atribut: chars** Private Static

Typ: String

Popis: Řetězec znaků, ze kterého metody generateRandomString generují náhodné znaky.

**Metoda: generateRandomString** Public Static

Popis: Vrátí řetězec náhodný alfanumerický řetězec o defaultní délce například 10 znaků. Záleží na tom, co je nastaveno v atributu LENGTH.

**Metoda: generateRandomString** Public Static

Popis: Stejné jako verze bez parametru, ale délku řetězce nastavíme parametrem len.

***Parametr: len***

Typ: int

Popis: Délka náhodného řetězce,