ActivityPlanner

Správa o realizácii projektu

Juraj Baráth
04.05.2018

Zámer projektu

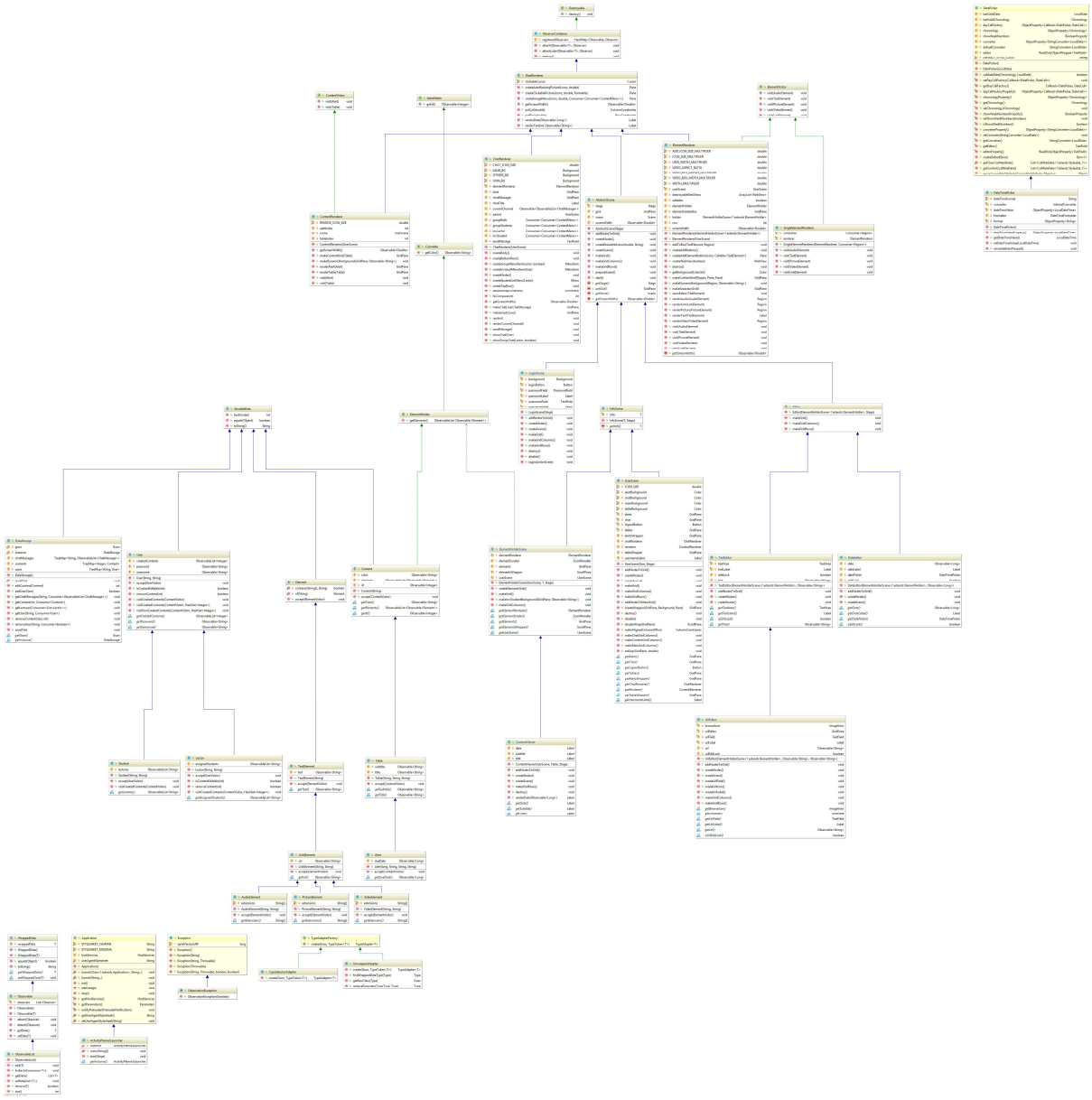
ActivityPlanner je softvér na organizáciu malých podujatí, ktorý slúži na plánovanie rôznych projektov. Vzhľadom na využitie softvéru na vzdelávacie účely, ku každému projektu bude patriť časť pre študentov a časť pre lektorov.

Najdôležitejšia vec pri organizácii je vzájomná komunikácia medzi jednotlivými členmi danej skupiny a pridávanie dôležitých informácii do priehľadného systému. ActivityPlanner na to bude používať informačné štítky a pripomienky. Informačné štítky sú označené farbami podľa kategórie. Tieto štítky sú pomenované a obsahujú krátke zhrnutie učiva a samotný obsah učiva. Zložitejšie vypracovaná verzia programu Activity Planner môže okrem textu učiva obsahovať aj tabuľky, grafy, linky, audio alebo videonahrávky. Štítky slúžia na zhromažďovanie informácií počas celého projektu. Activity planner bude obsahovať aj možnosť nastavenia pripomienok na potrebu vykonať časovo limitované činnosti. Napr. denné, týždenné, mesačné úlohy jednotlivých členov skupiny.

Praktické použitie: Lektor spustí program, a študenti sa do neho prihlásia. Program je možné aplikovať aj pri spolupráci viacerých škôl, napr. aj so zahraničnými partnerskými školami. Dôležité pritom je, aby študenti mohli medzi sebou komunikovať a vzájomne si pomáhať a to isté sa týka aj lektorov.

Aby lektori vedeli aké inštrukcie môžu používať pri nastavení serveru, budú mať prístup k manuálu. Lektori môžu v rámci programu pridať ďalších používateľov, alebo ich naopak vyradiť.

Class diagram



Kritéria Hodnotenia

- Zapuzdrenie a dedenie medzi vlastými triedami je možné vidieť na class diagramu.
- Program obsahuje dostatok komentáru, ako aj JavaDOC pre všetky metódy
- **Použité návrhové vzory:** Observer a Visitor
- Grafické používateľské rozhranie sa nachádza v balíku **gyurix.activityplanner.gui** a aplikačná logika sa nachádza v balíku **gyurix.activityplanner.core**. Sú úplne oddelené.
- Explicitné použitie viacniťovosti (multithreading) sa nachádza v triedy ElementRenderer a je použité na asynchrónne I/O operácie, ako napríklad načítanie obrázkov zo súboru / z webu.
- Používam generickosť v triede **WrappedData** a v jej extenziach (**Observable** a **ObservableList**)
- Používam lamda výrazov pri vlastných action handler-ov pripojený k tlačítkach v GUI.
- Používam referencií na metód v triede **LoginScreen**.
- Používam AspectJ na automatické uloženie zmien (súbor **AutoSaver.aj**).
- V aplikačnom logikes a nachádza vlastná výnimka ObservationException pre upozornenie klientský kód na nesprávne použitie triedu ObservableList
- Chytím výnimky pri I/O operáciach na výpis chyby do konzolu

Zoznam odovzdaných pracovných verzií

1.0:

- Initial release

1.1:

- Setup a whole maven building process
- Added AutoSaver aspect for automatically saving the configuration on every data change
- Added UML class hierarchy visualization for the core part
- Added documentation (README.MD with screenshots and JavaDoc)

1.1.1:

- Added JavaDOC zip generation to build process

1.1.2:

- Improved generated UML
- Added project realization document