# Zabezpečovací systém v starobinci

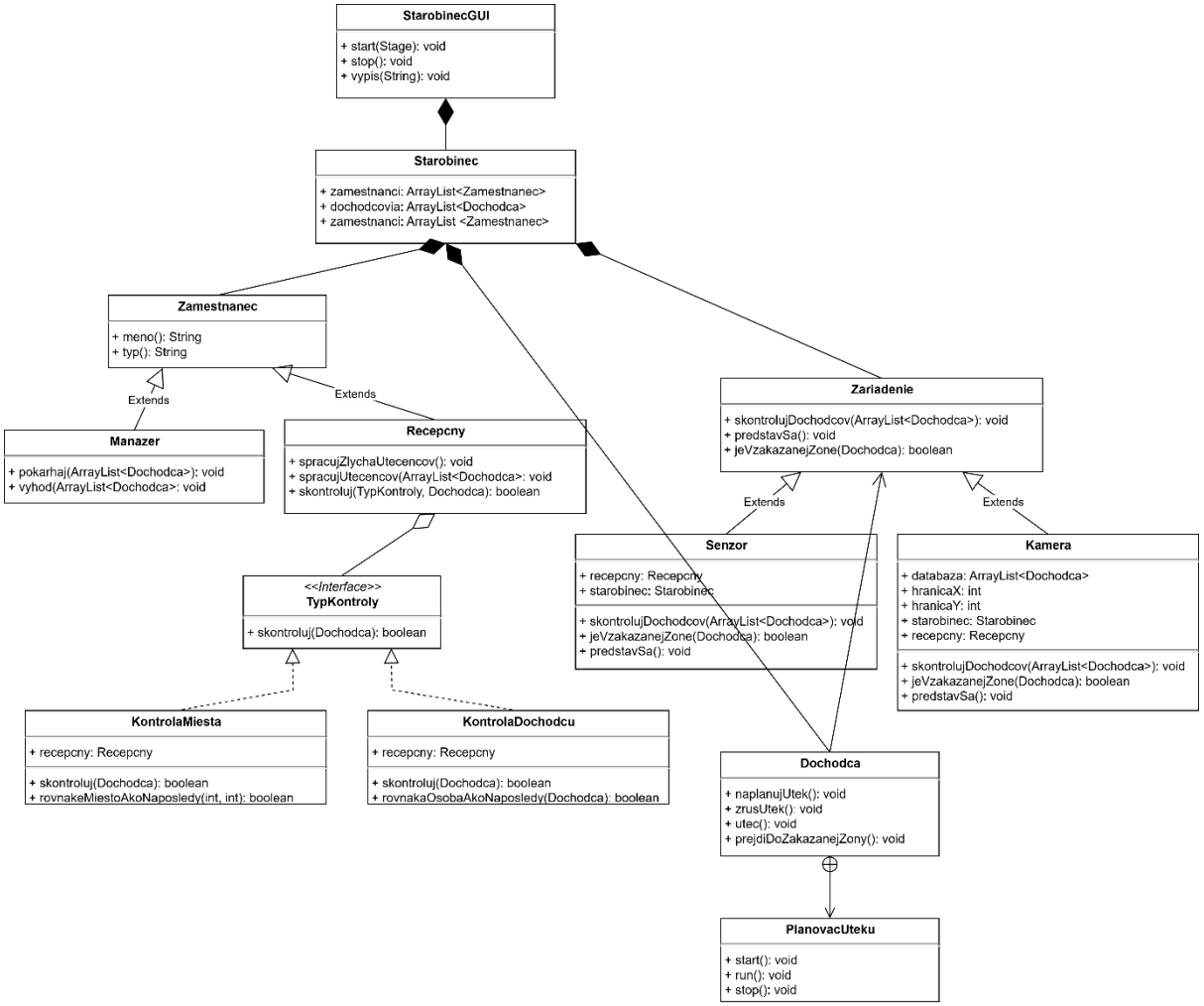
Jakub Šuran

FIIT STU - OOP

# Zámer projektu

Témou tohto projektu bude ukážka simulácie zabezpečovacieho systému v starobinci. Starobince sú vo všeobecnosti známe tým, že poskytujú domov pre starších ľudí, ktorým sa často stáva, že už majú nejaké psychické problémy alebo jednoducho kašlú na mravné zásady, čo môže vyústiť k nedodržovaniu pravidiel. Majitelia starobincov sa chcú zabezpečiť, chcú kontrolovať a sledovať pohyb všetkých dôchodcov, pre nich to ale vôbec nie je jednoduché, keďže nedokážu jednoducho vyfiltrovať všetkých svojich zamestnancov a dôchodcov. Práve na riešenie tohto problému dostanú vypracovaný dôveryhodný softvér, ktorý im pomôže oveľa jednoduchším spôsobom sledovať pohyb jednotlivých osôb. Systém bude spracovávať dáta z rôznych zabezpečovacích zariadení, kamerových systémov, senzorov pohybu. Kamery v starobinci budú disponovať najnovšími funkciami, ako napríklad rozpoznávanie tváre, rozlišovanie ľudí a zamestnancov a ostatných predmetov od dôchodcov s funkciou deep learning, budú schopné redukovať množstvo falošných poplachov. Všetky dostupné zariadenia budú odosielať svoje signály na recepciu, kde budú recepčné dohliadajúce na celý chod systému. Samozrejme, že softvér sa môže občas pomýliť, pre takéto prípady tu budú manažéri senior dómu, ktorí dostanú echo od pracovníkov z recepcie, keď sa vyskytne situácia, ktorú bude potrebné preriešiť manuálne.

# Diagram dôležitých tried



# Plnenie kritérií

## Hlavné kritériá

Program je funkčný a zodpovedá zadaniu a zámeru projektu, ktoré schválil vyučujúci, a zásadným požiadavkám vyučujúceho, ktoré vznikli počas realizácie projektu. Odovzdaný zdrojový kód zahŕňa všetky potrebné súbory a dá sa preložiť v prostredí Eclipse. Program obsahuje zmysluplné dedenie medzi vlastnými triedami s prekonávaním vlastných metód. V programe je použité zapuzdrenie. Program obsahuje dostatok komentáru na pochopenie kódu. Dokumentácia zodpovedá programu a obsahuje diagram tried. V projekte sú uplatnené objektovo-orientované mechanizmy, vhodné využitie dedenia v dvoch oddelených hierarchiách, polymorfizmu, zapuzdrenia a agregácie, kód je organizovaný do balíkov.

## Ďalšie kritériá

### *použitie návrhových vzorov okrem návrhového vzoru Singleton*

implementovaný návrhový vzor **Strategy** – vytvorený interface **TypKontroly**, ktorý implementujú triedy **KontrolaMiesta** a **KontrolaDochodcu**, ktoré sú využívané triedou **Recepcny**, pričom každá trieda vykonáva kontrolu jedinečným spôsobom

### *ošetrenie mimoriadnych stavov prostredníctvom vlastných výnimiek*

ošetrenie mimoriadneho stavu keď neexistuje žiadny dôchodca, pomocou vlastnej triedy **NonDochodcaException**, pričom sa používateľovi v GUI vypíše upozornenie a nebude sa dať pokračovať ďalej bez vytvorenia dôchodcu. Uplatnenie napríklad v triede Starobinec, riadok 112

### *poskytnutie grafického používateľského rozhrania oddelene od aplikačnej logiky a s aspoň časťou spracovateľov udalostí (handlers) vytvorenou manuálne*

grafické používateľské rozhranie implementované v triede **StarobinecGUI**

### *explicitné použitie viacniťovosti (multithreading) – spustenie vlastnej nite priamo alebo prostredníctvom API vyššej úrovne*

implementovaná trieda **PlanovacUteku** priamo pomocou implementovania rozhrania **Runnable**. Uplatnenie v triede Dochodca, riadok 36

### *explicitné použitie RTTI – napr. na zistenie typu objektu alebo vytvorenie objektu príslušného typu*

**RTTI** použité v triede **Starobinec** v metóde **vykonajKontrolu**, riadok 143

*použitie vhniezdených tried a rozhraní – počíta sa iba použitie v aplikačnej logike, nie v GUI*

implementovanie **vhniezdenej triedy** PlanovacUtoku v triede Dochodca, riadok 36

*použitie lambda výrazov alebo referencií na metódy (method references) – počíta sa iba použitie v aplikačnej logike, nie v GUI*

použité **lambda** výrazy – napríklad v triede Starobinec, riadok 114

*použitie serializácie*

použitá **serializácia** v triede **Starobinec** a **Dochodca**. Využité zachovávanie dát z predchádzajúcej verzie – zachovanie dôchodcov, ktorý zostali v predchádzajúcej verzii Starobinca.

# Zoznam hlavných verzií na GitHub classroom

1. **Commit: updated readme (14.4.2020 – 1:31)**
   ec6701074c07c1972ec8e35a1421db9b691c00e7
   - v tomto commite bola dokončená pracovná verzia programu
2. **Commit: added serialization, own exceptions, strategy design pattern, lambda (5.5.2020 20:02)**
   389549cf80b70648735f73f41998e373d784b895
   - v tomto commite bola pridaná serializácia, vlastné výnimky, návrhový vzor strategy, lambda výrazy
3. **Commit: multithreading + nested class (13.5.2020 12:26)**
   8a18059569ocb8205225547895d7b700623ae860
   - v tomto commite bola pridaná viacniťovosť a vzhniezdenie triedy
4. **Commit: added maintenance mode (13.5.2020 16:06)**
   86435252a7392c936a9bf3b3ec4fdf353b4a3049
   - v tomto commite bol pridaný mód údržby
5. **Posledný commit: added generated javadoc folder (25.5.2020 1:06)**
   3d2e87e809db366622d310748a96bf2bb039b814
   - v tomto poslednom commite pred finálnym, bol vložený javadoc

# Dodatočné informácie

Súbor v repozitári na GitHube obsahuje súbor README.md, ktorý je dostupný už od pracovnej verzie a obsahuje dôležité informácie ako základný popis a ovládanie programu, návod ako bezproblémovo spustiť program v Eclipse a uplatnenie základných OOP princípov