**Návrhové vzory - creational design patterns, structural design patterns, behavioral patterns**

**Návrhové vzory**

**Typické/obecné řešení** pro **časté a známé problémy** při vývoji SW.

Koncept/šablona který se musí předělat pro vlastní použití a řešení problému.

OOP návrhové vzory **typicky ukazují vztahy mezi třídy a objekty**.

Nejsou jen z programování, jsou také v každodenním životě

* Např. architektura (stavby)

**!! ALGORITMY – Nejsou návrhové vzory, ty řeší konkrétní problémy !!**

**ZÁKLADNÍ TYPY**

* **Creational patterns (vytvářející)**
  + **Vytváření objektů v systému**
    - Znovu použitelnost
  + Dynamická rozhodnutí za běhu programu
  + Příklady
    - Singleton, Factory method
* **Structural patterns (strukturální)**
  + **Uspořádání jednotlivých tříd nebo komponentů**
  + Důvody
    - Zpřehlednit systém
    - Strukturalizace kódu
  + Příklady
    - Decorator, Proxy
* **Behavioral patterns (chování)**
  + **Chování mezi objekty**
  + **Povinnosti objektů**
  + Objekty
    - Spolupráce mezi objekty
    - Spolupráce mezi skupiny objektů
  + Příklady
    - Observer, Command

**CREATIONAL PATTERNS**

**Factory method**

**Pokud objekty stejného typu vytvářeny na více místech redundantně**

* **Udělá se FACTORY pro typ těch objektů**
* **V ty se udělá metoda CREATE která bude tvořit Děti daného objektů**
  + **Vymizí redundance**

**Struktura**

**Diagram

Description automatically generated**

**A screen shot of a computer program

Description automatically generated with low confidence**

**Singleton**

Singleton je návrhový vzor, který umožňuje zajistit, **aby třída měla pouze jednu instanci**, a **zároveň poskytuje globální přístupový bod k této instanci**.

**Diagram

Description automatically generated**

**A screen shot of a computer program

Description automatically generated with medium confidence**

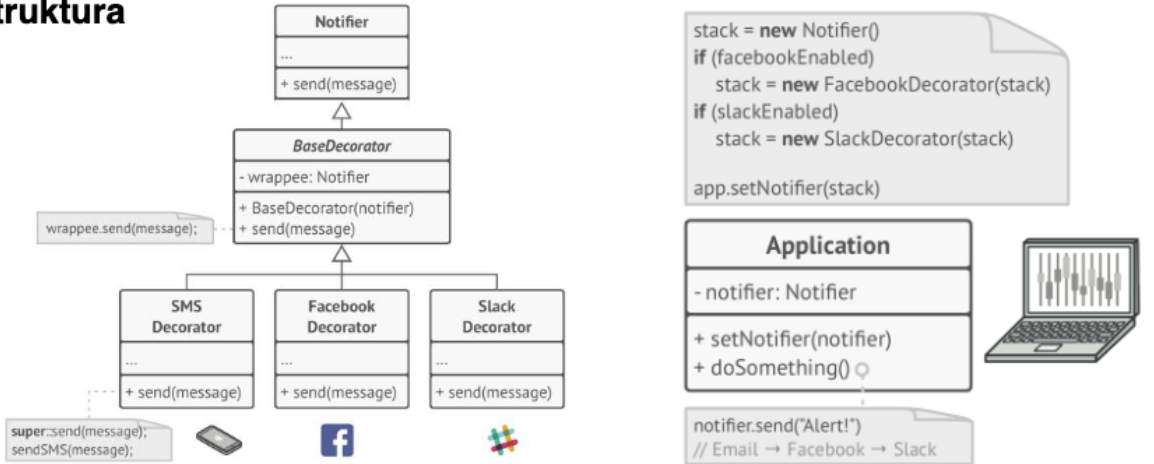
**Structural patterns**

Vysvětluje, jak dávat objekty a třídy dohromady do společných velkých struktur a zanechat flexibilitu a efektivitu.

**Decorator**

Umožňuje přidat **nové funkce objektům** tím**, že je umístí do speciálního “wrapperu” (obalu).**

****

****

**Proxy**

**Meziobjekt**, který je **mezi cílovém objektu a všech ostatních**

* **Ovládá přístup k cílovému objektu**
* umožňuje provést libovolné změny buď před nebo poté co se dotaz dostane skrz cílový objekt.

**Máme LIBRARY která executuje SQL příkazy**

* **V library není ošetřeno SQL INJECTION**
* **Pomocí proxy odfiltrujeme SQL INJECTION a následně spustíme metodu v LIBRARY**

**Struktura**

**A picture containing text

Description automatically generated**

**Kód**

**Graphical user interface, text

Description automatically generated**

**Behavioral patterns**

Zodpovídá za **komunikaci a přiřazování zodpovědností mezi objekty**

**Command**

Mění request do samostatného objektu

* objekt obsahuje všechny informace o requestu

**Toto umožňuje**

* přidat parametry metodám s různými requesty
* zpoždění splnění requestu
* vložení requestů do fronty.

**Diagram

Description automatically generatedStruktura**

**Ukázka kódu**

**A screenshot of a computer

Description automatically generated with medium confidence**

**Observer**

Vzor návrhu chování, který **umožňuje definovat funkci předplatného** = **upozornění více objektů na události, ke kterým dojde u objektu, který pozorují**.

**Struktura**

**Diagram

Description automatically generated**

**Ukázka kódu**

**Graphical user interface, text, application, chat or text message

Description automatically generated**

**Iterator**

* **Předpis jak se bude procházet datová struktura**