* **Design Problems** in applications pot fi rezolvate prin **Design Patterns**
* In Software Eengineering ne putem intalni de mai multe ori cu aceeasi Design Problem. Solutii sunt multe, dar unele dintre ele sunt mai preferate, caci sunt mai flexibile si mai bune deci.
* **Design pattern** - este o solutie practica, demonstrata, la o Design Problem, de catre software developers experimentati.
* Sunt in jur de 23 de design patterns des folosite.
* Design paterns sunt coneptuale. Ele ne ajuta sa nu trebuiasca sa cream noi o solutie de la 0, dar sa avem o solutie conceptuala deja creata si testata mai inainte de experti.
* **Pattern Language** - colectie de patterns legate de o anumita problema specifica
* De ex, cand cream un Joc vom folosi un anumit pattern language, cand vom crea o aplicatie de Banking, iar un alt apttern language
* Pattern Language e ales in dependenta de contextul in care lucram.

**Categories of patterns**

* **Creational Patterns** - pattern despre cum cream/clonam noi obiecte
* **Structural Patterns** - descrie cum obiectele sunt conectate unele cu altele. Aici apare si Decomposition si Generalization. Descrie si cum clasele si subclasele interactioneaza prin inheritance. In general, structurla patterns descriu cum lucreaza obiectele impreuna.
* **Behavioral patterns** - descrie cum obiectele focuseaza lucrul. Ele descriu cum fiecare obiect face o singura choesive function. Se focuseaza si pe cum obiectele independente lucreaza pentru a atinge un gol comun. De ex, la o cursa, fiecare participant are o echipa, care se ocupa si de masina,si toti lucreaza ca cursa sa fie castigata. Asa si lucreaza behaviorla patterns, caci el descrie cum mai multe obiecte ruleaza impreuna sa obtina un gol comun.