Programovanie v C#

Dedičnosť a polymorfizmus

Ing. Martina Szabóová, PhD.

Zopakujme si.

Čo je to trieda?

Zopakujme si.

Trieda je šablona (vzor), ktorý predpisuje
aké inštančné premenné a aké metódy majú objekty danej triedy
a čo sa udeje pri zavolaní týchto metód

Dedičnosť (Inheritance)

- Znovupoužiteľnosť (reusability) - používame to, čo už existuje
- Rozšíriteľnosť (extendibility) - rozširujeme funkcionalitu, máme zaručený typovú konzistenciu, nové dátove štruktúry, notácie, operácie

Dedičnosť (Inheritance)

- vytvárame nové triedy použitím vlastností už existujúcich tried
- pomenovanie odvodenia novej triedy z už existujúcej triedy sa nazýva dedičnosť
- nová triedu znovupoužíva, rozširuje alebo modifikuje správanie pôvodnej triedy
- pôvodná trieda sa nazýva : „**base**“ class príp. „super“ class alebo „parent“ class
- odvodená trieda : „**derived**“ class; „sub“ class alebo „child“ class

Dedičnosť (Inheritance)

- C# podporuje dedenie iba od jednej triedy
- je tranzitívne, t.j. ak D je odvodená trieda od C to od B to od A, tak potom členy triedy A sú dostupné triede D
- nie všetky členy triedy sú dediteľné odvodenými triedami:
  - statické konštruktory - inicializuje statické premenné
  - konštruktor - na vytvorenie novej inštancie triedy
  - finalizér (deštruktor) - na uvoľnenie miesta v pamäti po zániku objektu
- či sú ostatné členy triedy prístupné nastavujeme modifikátormi prístupu

Kľúčové slová

- Private členy - nie sú prístupné odvodeným triedym (výnimka nested triedy!)
- Protected členy - prístupné iba v odvodených triedach
- Internal členy - prístupné iba v tých triedach, ktoré sú umiestnené v rovnakej assembly ako base trieda
- Public členy - viditeľné v odvodených triedach a sú súčasťou public rozhraní odvodenej triedy

Method overriding vs new

- Odvodené triedy môžu prepísať zdedených členov
- **virtual -** aby bolo možné prepísať členy v base triede, je potrebné nastaviť ich
- predvolene nie sú virtual a nemôžu byť prepísané
- v niektorých prípadoch chceme aby odvodená trieda prepísala členy base triedy
- ak použijeme **new** - skryjeme metódy base triedy, používaju nové správanie v rámci odvodenej triedy

- členy base triedy označené **abstract,** označujú členy, ktoré majú byť v odvodenej triede prepísane
- abstraktné členy triedy musia byť uložené v abstraktnej triede
- **sealed -** od danej triedy nie je možné dediť ďalej

Dedičnosť

- dedičnosť sa vzťahuje len na triedy a rozhrania
- iné typy (štruktúry, enumeračné typy,...) nepodporujú dedičnosť

Polymorfizmus (Polymorphism)

- jeden názov = veľa foriem
- hlavné 2 typy:
  - subtyping (runtime)
  - parametrický (compile time) - generiká
  - ad-hoc - preťažovanie metód (overloading)
    - metódy majú rovnaký názov, ale rôzny popis (typ návratovej hodnoty, typ parametrov, poradie parametrov)
    - nemôžeme preťažiť metódu len zmenou návratovej hodnoty
    - výber metódy sa vykoná pri kompilovaní programu
  - coercion - casting - pretypovanie - napr. int na float

Subtyping

Dynamic/Runtime/Inclusion Polymorfizmus

- použitie jedného typu tam, kde sa očakáva použitie druhého typu
- metódy/funkcie napísane pre supertype budú fungovať aj na subtype
- ak trieda B je podtriedy triedy A, tak všade, kde sa očakáva A je možné zadať aj B
- late binding
- prepisovanie metód (overriding)
  - názov metód a ich popis (typ a počet parametrov) musí byť rovnaký
  - prekladač rozhodne počas behu programu, ktorú metódu zavolať (ak nie je žiadna k dispozícií vráti nám chybu)

Parametrický

Static/Compile Time Polymorfizmus

- early binding
- generiká
- typ ako parameter, explicitne alebo implicitne podľa potreby nahradíme premenné konkrétnymi typmi
- napr. zoznam

Ďakujem za pozornosť