Dokumentace javového kódu

Tomáš Pitner, upravil Marek Šabo

Dokumentace javových programů I

- Dokumentace je *nezbytnou součástí* javových programů.
- Existuje mnoho druhů dokumentací dle účelu:
  - instalační (pokyny pro nasazení produktu)
  - systémová (konfigurace, správa produktu)
  - uživatelská (pro užívání produktu)
  - vývojářská neboli programátorská (pro údržbu, rozšiřování a znovupoužití)

Dokumentace javových programů II

- Zde se budeme věnovat především dokumentaci *programátorské*.
- To znamená pro ty, kteří budou náš kód využívat ve svých programech, rozšiřovat jej, udržovat jej.
- Programátorské dokumentaci se říká *dokumentace API*, *javadoc*.
- Nejlepším příkladem je přímo dokumentace Java Core API

Pravidla komentování

Při psaní dokumentace dodržujeme tato pravidla:

- Jedná se hlavně o dokumentaci pro ostatní, tudíž prioritně dokumentujeme to, co je pro použití ostatními
- Dokumentujeme především celé třídy a jejich `public` a `protected` metody.
- Ostatní věci dle potřeby, například těžce pochopitelné nebo nelogické pasáže kódu.
- Dokumentaci píšeme *přímo do zdrojového kódu* ve *speciálních dokumentačních komentářích*.
- Dovnitř metod nepíšeme většinou *žádné* komentáře, ale můžeme (obtížné části).
- Ve výuce (ale často i v praxi) budou komentáře ve vašem projektu vynucovány nástrojem *checkstyle*.

Typy komentářů

- *řádkové* - od značky `//` do konce řádku, nepromítnou se do dokumentace

```
// inline comment, no javadoc generated
```

- *blokové* - mohou být na libovolném počtu řádků, nepromítnou se do dokumentace

```
/* block comment, no javadoc generated */
```

- *dokumentační* - libovolný počet řádků, promítnou se do dokumentace

```
/** documentary comment, javadoc generated */
```

- *Řádkové* a *blokové* komentáře by měly být pouze dočasné, ve výsledném kódu by měly být komentáře pouze *dokumentační*.

Nástroj `javadoc`

- Slouží ke strojovému generování dokumentace z dokumentačních komentářů a ze samotné struktury programu
- Dokumentace se vygeneruje do sady HTML souborů, takže to bude vypadat jako v Java Core API.
- Používá speciální značky se znakem zavináč, např. `@author`, v dokumentačních komentářích

Značky nástroje `javadoc`

- Javadoc používá značky vkládané do dokumentačních komentářů; hlavními jsou:

`@author`
: specifikuje autora

`@param`
: popisuje jeden parametr metody

`@return`
: popisuje co metoda vrací

`@throws`
: popisuje informace o výjimce, kterou metoda propouští ("vyhazuje")

a další…

Co dokumentujeme

- Dokumentujeme především *celé třídy*, zejména veřejné
- Jejich `public` a `protected` *metody*
- Lze dokumentovat i *celý balík* a to sepsáním souboru `package-info.html` v příslušném balíku
- V pořádných knihovnách (API) dokumentace k balíkům je

Komentář třídy

```
/**
 * This class represents a point in two dimensional space.
 *
 * @author Petr Novotny
 **/
 public class Vertex2D { ... }
```

Komentář metody I

- Zkrácený oficiální komentář metody toString():

```
/**
 * Returns a string representation of the object. In general, the
 * {@code toString} method returns a string that
 * "textually represents" this object. The result should ...
 *
 * @return  a string representation of the object.
 */
 public String toString() { ... }
```

- Část `{@code toString}` značí formátování, vypíše to `toString` neproporcionálním písmem.

Komentář metody II

```
/**
 * Returns the smaller of two int values.
 * If the arguments have the same value, the result is that same value.
 *
 * @param   a   an argument.
 * @param   b   another argument.
 * @return  the smaller of {@code a} and {@code b}.
 */
public int min(int a, int b) {
    return (a <= b) ? a : b;
}
```

Kde uvádíme dokumentační komentáře

- Dokumentační komentáře uvádíme:
  - Před hlavičkou *třídy* — pak komentuje třídu jako celek.

    - Před hlavičkou *metody* — pak komentuje příslušnou metodu.
- Doporučení Sun/Oracle k psaní dokumentačních komentářů — How to Write Doc Comments for the Javadoc Tool

Problémy dokumentování

- Komentáře lžou — změním kód a zapomenu upravit komentář

```
/**
 * Calculates velocity.
 */
System.out.println(triangle.getArea());
```

- Zbytečné komentáře

```
private int size; // creates size
```

- Ideální je psát kód a názvy tříd a metod tak, aby se komentáře ani nemusely číst