# Styl „Nadpis1“ pro kapitoly

## Styl „Nadpis2“ pro podkapitoly, kdyby je někdo náhodou potřeboval

Styl „Normální“ pro běžný odstavcový text

Styl „Počítačový kód“ pro mimořádně zajímavé části kódu.

# Co psát do dokumentace?

Představte si sebe v situaci, kdy jste obeznámeni se zadáním semestrální práce a najednou se vám do ruky dostane projekt vašeho kamaráda v nějaké fázi rozpracovanosti. Vaším úkolem je projekt dokončit, tj. navázat na to, co je již k dispozici. Co by vám vaši úlohu usnadnilo?

Mít JavaDoc dokumentaci API je jistě užitečné, ale možná jste již zjistili, že se v JavaDoc dokumentaci snadno ztratíte, když nevíte, co přesně hledáte. Uveďme si příklad. Víte, že potřebujete upravit vykreslení grafů, ale v JavaDoc dokumentaci vidíte hned 15 tříd. Která z nich se asi tak stará o vykreslení grafů? Něco lze odhadnout z názvů tříd. Někdy vám nezbyde nic jiného než si projít popis hned několika tříd. A někdy se řešení ani nedoberete jednoduše proto, že třída, která se o vykreslení grafů stará, je privátní / interní, takže se v dokumentaci API vůbec neobjeví. Pak zbývá jen zkusit si tipnout, jak by se mohla jmenovat metoda / třída, která se o vykreslení grafu stará (asi by mohla mít v názvu „graph“), a prohledat zdrojové kódy na výskyt tohoto jména. Dost pracné, co říkáte? Nebylo by super mít nějakou další pomocnou dokumentaci, která by vám pomohla s prvotní orientací v projektu? A právě k tomu slouží tato dokumentace.

Dokumentace by měla obsahovat:

* stručný ale výstižný a přehledný popis implementovaného řešení (např. „hodnoty ABC za posledních 100 sekund se zobrazují spojnicovým grafem; graf se plynule překresluje, jak přicházejí nové hodnoty ABC“) včetně toho, jak je to na nejvyšší úrovni dekomponováno do tříd a balíčků (např. „řešení je postaveno na softwarovém vzoru MVVM, … Nejdůležitějšími třídami ViewModel jsou XYZ, která se stará o veškerou interakci uživatele pro výběr dat pro zobrazení v grafech, … “)
* popis zavedených opodstatněných omezení a zjednodušení (např. „je použito sekvenční vyhledávání prvku v poli, protože typická délka pole nepřesáhne 10 prvků; pokud by se v budoucnosti pracovalo s podstatně většími poli, lze zvážit použití binárního vyhledání“)
* popis klíčových algoritmů za účelem objasnění chování, přičemž tento popis, často zapsán v nějakém pseudokódu, je na vyšší úrovni abstrakce, než je pak vlastní kód v programovacím jazyce. Rozhodně nekopírujte kusy kódy
* popis vytvoření, instalace a spuštění aplikace, pokud toto není intuitivní, nebo apriori dáno
* popis ovládání aplikace, pokud toto není intuitivní, nebo apriori dáno
* stručný popis dosud neopravených nedostatků a popis možného rozšíření do budoucna

Nezapomeňte rovněž vyplnit údaje v záhlaví! Celkový počet hodin = počet hodin soustředěné práce, tj. nezapočtou se do toho přestávky na oběd, večeři, vyřízení telefonu apod.