



BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INTELLIGENT SYSTEMS

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

CONTINUOUS INTEGRATION AND AUTOMATED CODE REVIEW IN OPEN SOURCE PROJECTS

PRŮBĚŽNÁ INTEGRACE A AUTOMATIZOVANÁ KONTROLA KÓDU V PROJEKTECH S OTEVŘENÝM
ZDROJOVÝM KÓDEM

BACHELOR'S THESIS

BAKALÁŘSKÁ PRÁCE

AUTHOR

AUTOR PRÁCE

ADRIÁN TÓTH

SUPERVISOR

VEDOUCÍ PRÁCE

Ing. LENKA TUROŇOVÁ,

BRNO 2017

Abstract

Do tohoto odstavce bude zapsán výtah (abstrakt) práce v anglickém jazyce.

Abstrakt

Do tohoto odstavce bude zapsán výtah (abstrakt) práce v českém (slovenském) jazyce.

Keywords

Sem budou zapsána jednotlivá klíčová slova v anglickém jazyce, oddělená čárkami.

Klíčová slova

Sem budou zapsána jednotlivá klíčová slova v českém (slovenském) jazyce, oddělená čárkami.

Reference

TÓTH, Adrián. *Continuous Integration and Automated Code Review in Open Source Projects*. Brno, 2017. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Lenka Turoňová,

Continuous Integration and Automated Code Review in Open Source Projects

Declaration

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana X... Další informace mi poskytli... Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....

Adrián Tóth
October 22, 2017

Acknowledgements

V této sekci je možno uvést poděkování vedoucímu práce a těm, kteří poskytli odbornou pomoc (externí zadavatel, konzultant, apod.).

Contents

1	Introduction	2
2	Continuous Integration & Automated Code Review	3
2.1	Continuous Integration	3
2.1.1	Demands of continuous integration	3
2.1.2	Stages of continuous integration	4
2.2	Automated Code Review	4
2.3	Musíme mít co říci	4
2.4	Musíme vědět, komu to chceme říci	4
2.5	Musíme si dokonale promyslet obsah	4
2.6	Musíme psát strukturovaně	5
3	Několik formálních pravidel	6
4	Nikdy to nebude naprosto dokonalé	7
5	Typografické a jazykové zásady	8
5.1	Co to je normovaná stránka?	9
6	Závěr	11
	Bibliography	12
A	Jak pracovat s touto šablonou	13

Chapter 1

Introduction

In these days, Continuous Integration (CI) is more often used in larger projects, where multiple developers are working on one and the same software product. This process ensures fast software development, called eXtreme Programming (XP), known as agile software development methodology. The methodology is mainly used to accelerate the development, nevertheless, development of software may be disrupted in various other ways. Nowadays, although this type of software development has many disadvantages, it is still much more often used on larger projects. The progress of the development may not be reached with a continuous integration which guarantees less disorder and failures. You may also know that the continuous integration is a part of the following open-source projects e.g. Facebook, Twitter, Mozilla. These projects use one of many famous continuous integration service Travis CI. Excluding Travis CI, there are plenty of other continuous integration services you may heard about, such as Jenkins, TeamCity, CircleCI, GitLab CI, Codeship and so on.

The software development process requires many code checking tools after every single code change in the source code. For as much as with every single change of code, there is a possibility to add, fix, derange or deteriorate any parts of the software product. These tools provide an automated code review and they afford a quick feedback by which they try to prevent these code impairments. Feedback about his adjustment is sent to the developer, who has made the change in the code. The automated process which provides the code review does not bother with executing a huge amount of tests. Above mentioned process is conducted via continuous integration server, which compiles the code, runs scripts and tests. The results are aggregated and the feedback is given to the developer who has made this code change. Continuous integration server is invoked every single time after any change is fetched in the source code and he had to execute the stated acts which are predefined. In next chapters we describe in details how does this workflow work and what steps are required to run.

The essence of this work is about the basics of continuous integration and its fundamentals. This thesis attempts to explain how the fundamentals of continuous integration and automated code review work. It describes how it is integrated to the software development, and how it works on an extensive project nowadays. Examples will be based on open-source project e.g. ManageIQ, which is a cloud manager founded by RedHat. The development process of the ManageIQ rests in agility and stability of the progress. These main factors of the development process could not be reached without a quick feedback to the developers working on project about their changes, that are submitted to the software product.

Chapter 2

Continuous Integration & Automated Code Review

2.1 Continuous Integration

Continuous integration has a key role in the software development process consisting of few certain unavoidable steps which will be described later in next subsections. Everything begins at the moment, when one developer, who has made changes in the source code of the software product, is trying to commit them into the software product. The process of continuous integration has begun at this point and lasts until feedback is sent back to the developer. These stages of continuous integrations are proceed every time after the CI server has detected any change in a version control repository. This automations has lots of benefits which are necessary to keep up the software product without any kind of defects. Many of them are detected in time and reported back to the developer as a corrupted source code. Not a few developers may think that the continuous integration is only about compiling the source code and launching tests. In the next subsections we will be guided through the steps of continuous integration and also we will describe these individual phases in details. [TODO: add image].

2.1.1 Demands of continuous integration

The minimal requirements for a good software development of a project, where are multiple developers working on the same project, is a version control repository and a continuous integration server. Version control system guarantee a software configuration management which is required for the continuous integration. The continuous integration could not be deployed without any version control system. The meaning of the version control system is very important. You cannot manage changes, that developers had made in the source code, without any version control system. Version control system has a very positive impact on the developing project. The system offers a history of changes, which may be highly useful if any rollback is desired. Besides the history of changes, this system may save more other information about the source code e.g. who did the change, etc. In addition, the version control system is representing a one primary source for the project source codes. This type of project setup is much more often used these days rather than in the past. Nearly every project has its own version control system deployed which is provided by any repository hosting service. [TODO: too many occurrences of vers.contr.sys.]

Continuous integration server has a one huge advantage, that's why it is highly recommended. It depends on you, how do you deploy a continuous integration server. With continuous integration server, you do not have to bother with such many scripts for the automation. Nevertheless, as you decide how would establish the continuous integration server, the system must contains these features. To facilitate the process of continuous integration, the system must support services as polling version control system, retention of build history, launching predefined steps such as scripts and tests. Furthermore, the system should offer an opportunity to send a feedback back to the developers. This server executes a series of actions or steps taken in order to achieve a particular end of continuous integration. The next subsection will determine and state these fundamental steps of the continuous integration scenario, describe and illustrate them in details.

2.1.2 Stages of continuous integration

1. Aaa.
2. Bbb.
3. Ccc.
4. Ddd.

2.2 Automated Code Review

Abcd.

2.3 Musíme mít co říci

Dalším důležitým předpokladem dobrého psaní je *psát pro někoho*. Píšeme-li si poznámky sami pro sebe, píšeme je jinak než výzkumnou zprávu, článek, diplomovou práci, knihu nebo dopis. Podle předpokládaného čtenáře se rozhodneme pro způsob psaní, rozsah informace a míru detailů.

2.4 Musíme vědět, komu to chceme říci

Dalším důležitým předpokladem dobrého psaní je psát pro někoho. Píšeme-li si poznámky sami pro sebe, píšeme je jinak než výzkumnou zprávu, článek, diplomovou práci, knihu nebo dopis. Podle předpokládaného čtenáře se rozhodneme pro způsob psaní, rozsah informace a míru detailů.

2.5 Musíme si dokonale promyslet obsah

Musíme si dokonale promyslet a sestavit obsah sdělení a vytvořit pořadí, v jakém chceme čtenáři své myšlenky prezentovat. Jakmile víme, co chceme říci a komu, musíme si rozvrhnout látku. Ideální je takové rozvržení, které tvoří logicky přesný a psychologicky stravitelný celek, ve kterém je pro všechno místo a jehož jednotlivé části do sebe přesně zapadají. Jsou jasné všechny souvislosti a je zřejmé, co kam patří.

Abychom tohoto cíle dosáhli, musíme pečlivě organizovat látku. Rozhodneme, co budou hlavní kapitoly, co podkapitoly a jaké jsou mezi nimi vztahy. Diagramem takové organizace

je graf, který je velmi podobný stromu, ale ne řetězci. Při organizaci látky je stejně důležitá otázka, co do osnovy zahrnout, jako otázka, co z ní vypustit. Příliš mnoho podrobností může čtenáře právě tak odradit jako žádné detaily.

Výsledkem této etapy je osnova textu, kterou tvoří sled hlavních myšlenek a mezi ně zařazené detaily.

2.6 Musíme psát strukturovaně

Musíme začít psát strukturovaně a současně pracujeme na co nejsrozumitelnější formě, včetně dobrého slohu a dokonalého značení. Máme-li tedy myšlenku, představu o budoucím čtenáři, cíl a osnovu textu, můžeme začít psát. Při psaní prvního konceptu se snažíme zaznamenat všechny své myšlenky a názory vztahující se k jednotlivým kapitolám a podkapitolám. Každou myšlenku musíme vysvětlit, popsat a prokázat. Hlavní myšlenku má vždy vyjadřovat hlavní věta a nikoliv věta vedlejší.

I k procesu psaní textu přistupujeme strukturovaně. Současně s tím, jak si ujasňujeme strukturu písemné práce, vytváříme kostru textu, kterou postupně doplňujeme. Využíváme ty prostředky DTP programu, které podporují strukturovanou stavbu textu (předdefinované typy pro nadpisy a bloky textu).

Chapter 3

Několik formálních pravidel

Naším cílem je vytvořit jasný a srozumitelný text. Vyjadřujeme se proto přesně, píšeme dobrou češtinou (nebo zpravidla angličtinou) a dobrým slohem podle obecně přijatých zvyklostí. Text má upravit čtenáři cestu k rychlému pochopení problému, předvídat jeho obtíže a předcházet jim. Dobrý sloh předpokládá bezvadnou gramatiku, správnou interpunkci a vhodnou volbu slov. Snažíme se, aby náš text nepůsobil příliš jednotvárně používáním malého výběru slov a tím, že některá zvláště oblíbená slova používáme příliš často. Pokud používáme cizích slov, je samozřejmým předpokladem, že známe jejich přesný význam. Ale i českých slov musíme používat ve správném smyslu. Např. platí jistá pravidla při používání slova *zřejmě*. Je *zřejmě* opravdu zřejmé? A přesvědčili jsme se, zda to, co je *zřejmě* opravdu platí? Pozor bychom si měli dát i na příliš časté používání zvrtného se. Například obratu *dokázalo se*, že... zásadně nepoužíváme. Není špatné používat autorského *my*, tím předpokládáme, že něco řešíme, nebo například zobecňujeme spolu se čtenářem. V kvalifikačních pracích použijeme autorského *já* (například když vymezujeme podíl vlastní práce vůči převzatému textu), ale v běžném textu se nadměrné používání první osoby jednotného čísla nedoporučuje.

Za pečlivý výběr stojí i symbolika, kterou používáme ke *značení*. Máme tím na mysli volbu zkratk a symbolů používaných například pro vyjádření typů součástí, pro označení hlavních činností programu, pro pojmenování ovládacích kláves na klávesnici, pro pojmenování proměnných v matematických formulích a podobně. Výstižné a důsledné značení může čtenáři při četbě textu velmi pomoci. Je vhodné uvést seznam značení na začátku textu. Nejen ve značení, ale i v odkazech a v celkové tiskové úpravě je důležitá důslednost.

S tím souvisí i pojem z typografie nazývaný *vyznačování*. Zde máme na mysli způsob sazby textu pro jeho zvýraznění. Pro zvolené značení by měl být zvolen i způsob vyznačování v textu. Tak například klávesy mohou být umístěny do obdélníčku, identifikátory ze zdrojového textu mohou být vypisovány **písmem typu psací stroj** a podobně.

Uvádíme-li některá fakta, neskrýváme jejich původ a náš vztah k nim. Když něco tvrdíme, vždycky výslovně uvedeme, co z toho bylo dokázáno, co teprve bude dokázáno v našem textu a co přebíráme z literatury s uvedením odkazu na příslušný zdroj. V tomto směru nenecháváme čtenáře nikdy na pochybách, zda jde o myšlenku naši nebo převzatou z literatury.

Nikdy neplýtváme čtenářovým časem výkladem triviálních a nepodstatných informací. Neuvádíme rovněž několikrát totéž jen jinými slovy. Při pozdějších úpravách textu se nám může některá dříve napsaná pasáž jevit jako zbytečně podrobná nebo dokonce zcela zbytečná. Vypuštění takové pasáže nebo alespoň její zestručnění přispěje k lepší čitelnosti práce! Tento krok ale vyžaduje odvahu zahodit čas, který jsme jejímu vytvoření věnovali.

Chapter 4

Nikdy to nebude naprosto dokonalé

Když jsme už napsali vše, o čem jsme přemýšleli, uděláme si den nebo dva dny volna a pak si přečteme sami rukopis znovu. Uděláme ještě poslední úpravy a skončíme. Jsme si vědomi toho, že vždy zůstane něco nedokončeno, vždy existuje lepší způsob, jak něco vysvětlit, ale každá etapa úprav musí být konečná.

Chapter 5

Typografické a jazykové zásady

Při tisku odborného textu typu *technická zpráva* (anglicky *technical report*), ke kterému patří například i text kvalifikačních prací, se často volí formát A4 a často se tiskne pouze po jedné straně papíru. V takovém případě volte levý okraj všech stránek o něco větší než pravý – v tomto místě budou papíry svázány a technologie vazby si tento požadavek vynucuje. Při vazbě s pevným hřbetem by se levý okraj měl dělat o něco širší pro tlusté svazky, protože se stránky budou hůře rozevírat a levý okraj se tak bude oku méně odhalovat.

Horní a spodní okraj volte stejně veliký, případně potištěnou část posuňte mírně nahoru (horní okraj menší než dolní). Počítejte s tím, že při vazbě budou okraje mírně oříznuty.

Pro sazbu na stránku formátu A4 je vhodné používat pro základní text písmo stupně (velikosti) 11 bodů. Volte šířku sazby 15 až 16 centimetrů a výšku 22 až 23 centimetrů (včetně případných hlaviček a patiček). Proklad mezi řádky se volí 120 procent stupně použitého základního písma, což je optimální hodnota pro rychlost čtení souvislého textu. V případě použití systému LaTeX ponecháme implicitní nastavení. Při psaní kvalifikační práce se řiďte příslušnými závaznými požadavky.

Stupeň písma u nadpisů různé úrovně volíme podle standardních typografických pravidel. Pro všechny uvedené druhy nadpisů se obvykle používá polotučné nebo tučné písmo (jednotně buď všude polotučné nebo všude tučné). Proklad se volí tak, aby se následující text běžných odstavců sázel pokud možno na *pevný rejstřík*, to znamená jakoby na linky s předem definovanou a pevnou roztečí.

Uspořádání jednotlivých částí textu musí být přehledné a logické. Je třeba odlišit názvy kapitol a podkapitol – píšeme je malými písmeny kromě velkých začátečních písmen. U jednotlivých odstavců textu odsazujeme první řádek odstavce asi o jeden až dva čtverčíky (vždy o stejnou, předem zvolenou hodnotu), tedy přibližně o dvě šířky velkého písmene M základního textu. Poslední řádek předchozího odstavce a první řádek následujícího odstavce se v takovém případě neoddělují svislou mezerou. Proklad mezi těmito řádky je stejný jako proklad mezi řádky uvnitř odstavce.

Při vkládání obrázků volte jejich rozměry tak, aby nepřesáhly oblast, do které se tiskne text (tj. okraje textu ze všech stran). Pro velké obrázky vyčleňte samostatnou stránku. Obrázky nebo tabulky o rozměrech větších než A4 umístěte do písemné zprávy formou skládanky vřité do přílohy nebo vložené do záložek na zadní desce.

Obrázky i tabulky musí být pořadově očíslovány. Číslování se volí buď průběžné v rámci celého textu, nebo – což bývá praktičtější – průběžné v rámci kapitoly. V druhém případě se číslo tabulky nebo obrázku skládá z čísla kapitoly a čísla obrázku/tabulky v rámci kapitoly – čísla jsou oddělena tečkou. Čísla podkapitol nemají na číslování obrázků a tabulek žádný vliv.

Tabulky a obrázky používají své vlastní, nezávislé číselné řady. Z toho vyplývá, že v odkazech uvnitř textu musíme kromě čísla udát i informaci o tom, zda se jedná o obrázek či tabulku (například “... viz tabulka 2.7 ...”). Dodržování této zásady je ostatně velmi přirozené.

Pro odkazy na stránky, na čísla kapitol a podkapitol, na čísla obrázků a tabulek a v dalších podobných příkladech využíváme speciálních prostředků DTP programu, které zajistí vygenerování správného čísla i v případě, že se text posune díky změnám samotného textu nebo díky úpravě parametrů sazby. Příkladem takového prostředku v systému LaTeX je odkaz na číslo odpovídající umístění značky v textu, například návěští (`\ref{navesti}`) – podle umístění návěští se bude jednat o číslo kapitoly, podkapitoly, obrázku, tabulky nebo podobného číslovaného prvku), na stránku, která obsahuje danou značku (`\pageref{navesti}`), nebo na literární odkaz (`\cite{identifikator}`).

Rovnice, na které se budeme v textu odvolávat, opatříme pořadovými čísly při pravém okraji příslušného řádku. Tato pořadová čísla se píší v kulatých závorkách. Číslování rovnic může být průběžné v textu nebo v jednotlivých kapitolách.

Jste-li na pochybách při sazbě matematického textu, snažte se dodržet způsob sazby definovaný systémem LaTeX. Obsahuje-li vaše práce velké množství matematických formulí, doporučujeme dát přednost použití systému LaTeX.

Mezeru neděláme tam, kde se spojují číslice s písmeny v jedno slovo nebo v jeden znak – například *25krát*.

Členicí (interpunkční) znaménka tečka, čárka, středník, dvojtečka, otazník a vykřičník, jakož i uzavírací závorky a uvozovky se přimykají k předcházejícímu slovu bez mezery. Mezera se dělá až za nimi. To se ovšem netýká desetinné čárky (nebo desetinné tečky). Otevírací závorka a přední uvozovky se přimykají k následujícímu slovu a mezera se vynechává před nimi – (takto) a “takto”.

Pro spojovací a rozdělovací čárku a pomlčku nepoužíváme stejný znak. Pro pomlčku je vyhrazen jiný znak (delší). V systému TeX (LaTeX) se spojovací čárka zapisuje jako jeden znak “pomlčka” (například “Brno–město”), pro sázení textu ve smyslu intervalu nebo dvojic, soupeřů a podobně se ve zdrojovém textu používá dvojice znaků “pomlčka” (například “zápas Sparta – Slavie”; “cena 23–25 korun”), pro výrazné oddělení části věty, pro výrazné oddělení vložené věty, pro vyjádření nevyslovené myšlenky a v dalších situacích (viz Pravidla českého pravopisu) se používá nejdelší typ pomlčky, která se ve zdrojovém textu zapisuje jako trojice znaků “pomlčka” (například “Další pojem — jakkoliv se může zdát nevýznamný — bude neformálně definován v následujícím odstavci.”). Při sazbě matematického mínus se při sazbě používá rovněž odlišný znak. V systému TeX je ve zdrojovém textu zapsán jako normální mínus (tj. znak “pomlčka”). Sazba v matematickém prostředí, kdy se vzoreček uzavírá mezi dolary, zajistí vygenerování správného výstupu.

Lomítko se píše bez mezer. Například školní rok 2008/2009.

Pravidla pro psaní zkratk jsou uvedena v Pravidlech českého pravopisu [1]. I z jiných důvodů je vhodné, abyste tuto knihu měli po ruce.

5.1 Co to je normovaná stránka?

Pojem *normovaná stránka* se vztahuje k posuzování objemu práce, nikoliv k počtu vytištěných listů. Z historického hlediska jde o počet stránek rukopisu, který se psal psacím strojem na speciální předtištěné formuláře při dodržení průměrné délky řádku 60 znaků a při 30 řádcích na stránku rukopisu. Vzhledem k zápisu korekturních značek se používalo řádkování 2 (ob jeden řádek). Tyto údaje (počet znaků na řádek, počet řádků a proklad

mezi nimi) se nijak nevztahují ke konečnému vytištěnému výsledku. Používají se pouze pro posouzení rozsahu. Jednou normovanou stránkou se tedy rozumí $60 \cdot 30 = 1800$ znaků. Obrázky zařazené do textu se započítávají do rozsahu písemné práce odhadem jako množství textu, které by ve výsledném dokumentu potisklo stejně velkou plochu.

Orientační rozsah práce v normostranách lze v programu Microsoft Word zjistit pomocí funkce *Počet slov* v menu *Nástroje*, když hodnotu *Znaky (včetně mezer)* vydělíte konstantou 1800. Do rozsahu práce se započítává pouze text uvedený v jádru práce. Části jako abstrakt, klíčová slova, prohlášení, obsah, literatura nebo přílohy se do rozsahu práce nepočítají. Je proto nutné nejdříve označit jádro práce a teprve pak si nechat spočítat počet znaků. Přibližný rozsah obrázků odhadnete ručně. Podobně lze postupovat i při použití OpenOffice. Při použití systému LaTeX pro sazbu je situace trochu složitější. Pro hrubý odhad počtu normostran lze využít součet velikostí zdrojových souborů práce podělený konstantou cca 2000 (normálně bychom dělili konstantou 1800, jenže ve zdrojových souborech jsou i vyznačovací příkazy, které se do rozsahu nepočítají). Pro přesnější odhad lze pak vyextrahovat holý text z PDF (např. metodou cut-and-paste nebo *Save as Text...*) a jeho velikost podělit konstantou 1800.

Chapter 6

Závěr

Závěrečná kapitola obsahuje zhodnocení dosažených výsledků se zvlášť vyznačeným vlastním přínosem studenta. Povinně se zde objeví i zhodnocení z pohledu dalšího vývoje projektu, student uvede náměty vycházející ze zkušeností s řešeným projektem a uvede rovněž návaznosti na právě dokončené projekty.

Bibliography

- [1] Hlavsa, Z.; et al.: *Pravidla českého pravopisu*. Academia. 2005. ISBN 80-200-1327-X.
- [2] Rábová, Z.; Hanáček, P.; Peringer, P.; et al.: *Užitečné rady pro psaní odborného textu*. FIT VUT v Brně. November 2008. [Online; navštíveno 12.05.2015].
Retrieved from: http://www.fit.vutbr.cz/info/statnice/psani_textu.html

Appendix A

Jak pracovat s touto šablonou

V této kapitole je uveden popis jednotlivých částí šablony, po kterém následuje stručný návod, jak s touto šablonou pracovat.

Jedná se o přechodnou verzi šablony. Nová verze bude zveřejněna do konce roku 2017 a bude navíc obsahovat nové pokyny ke správnému využití šablony, závazné pokyny k vypracování bakalářských a diplomových prací (rekapitulace pokynů, které jsou dostupné na webu) a nezávazná doporučení od vybraných vedoucích, která již teď najdete na webu (viz odkazy v souboru s literaturou). Jediné soubory, které se v nové verzi změní, budou `projekt-01-kapitoly-chapters.tex` a `projekt-30-prilohy-appendices.tex`, jejichž obsah každý student vymaže a nahradí vlastním. Šablonu lze tedy bez problémů využít i v současné verzi.

Popis částí šablony

Po rozbalení šablony naleznete následující soubory a adresáře:

bib-styles Styly literatury (viz níže).

obrazky-figures Adresář pro Vaše obrázky. Nyní obsahuje `placeholder.pdf` (tzv. TODO obrázek, který lze použít jako pomůcku při tvorbě technické zprávy), který se s prací neodevzdává. Název adresáře je vhodné zkrátit, aby byl jen ve zvoleném jazyce.

template-fig Obrázky šablony (znak VUT).

fitthesis.cls Šablona (definice vzhledu).

Makefile Makefile pro překlad, počítání normostran, sbalení apod. (viz níže).

projekt-01-kapitoly-chapters.tex Soubor pro Váš text (obsah nahraďte).

projekt-20-literatura-bibliography.bib Seznam literatury (viz níže).

projekt-30-prilohy-appendices.tex Soubor pro přílohy (obsah nahraďte).

projekt.tex Hlavní soubor práce – definice formálních částí.

Výchozí styl literatury (czechiso) je od Ing. Martínka, přičemž anglická verze (englishiso) je jeho překladem s drobnými modifikacemi. Oproti normě jsou v něm určité

odlišnosti, ale na FIT je dlouhodobě akceptován. Alternativně můžete využít styl od Ing. Radima Loskota nebo od Ing. Radka Pyšného¹. Alternativní styly obsahují určitá vylepšení, ale zatím nebyly řádně otestovány větším množstvím uživatelů. Lze je považovat za beta verze pro zájemce, kteří svoji práci chtějí mít dokonalou do detailů a neváhají si nastudovat detaily správného formátování citací, aby si mohli ověřit, že je vysázený výsledek v pořádku.

Makefile kromě překladu do PDF nabízí i další funkce:

- přejmenování souborů (viz níže),
- počítání normostran,
- spuštění vlny pro doplnění nezlomitelných mezer,
- sbalení výsledku pro odeslání vedoucímu ke kontrole (zkontrolujte, zda sbalí všechny Vámi přidané soubory, a případně doplňte).

Nezapomeňte, že vlna neřeší všechny nezlomitelné mezery. Vždy je třeba manuální kontrola, zda na konci řádku nezůstalo něco nevhodného – viz Internetová jazyková příručka².

Pozor na číslování stránek! Pokud má obsah 2 strany a na 2. jsou jen “Přílohy” a “Seznam příloh” (ale žádná příloha tam není), z nějakého důvodu se posune číslování stránek o 1 (obsah “nesedí”). Stejný efekt má, když je na 2. či 3. stránce obsahu jen “Literatura” a je možné, že tohoto problému lze dosáhnout i jinak. Řešení je několik (od úpravy obsahu, přes nastavení počítadla až po sofistikovanější metody). **Před odevzdáním proto vždy přezkontrolujte číslování stran!**

Doporučený postup práce se šablonou

1. **Zkontrolujte, zda máte aktuální verzi šablony.** Máte-li šablonu z předchozího roku, na stránkách fakulty již může být novější verze šablony s aktualizovanými informacemi, opravenými chybami apod.
2. **Zvolte si jazyk,** ve kterém budete psát svoji technickou zprávu (česky, slovensky nebo anglicky) a svoji volbu konzultujte s vedoucím práce (nebyla-li dohodnuta předem). Pokud Vámi zvoleným jazykem technické zprávy není čeština, nastavte příslušný parametr šablony v souboru `projekt.tex` (např.: `documentclass[english]{fitthesis}`) a přeložte prohlášení a poděkování do angličtiny či slovenštiny.
3. **Přejmenujte soubory.** Po rozbalení je v šabloně soubor `projekt.tex`. Pokud jej přeložíte, vznikne PDF s technickou zprávou pojmenované `projekt.pdf`. Když vedoucímu více studentů pošle `projekt.pdf` ke kontrole, musí je pracně přejmenovávat. Proto je vždy vhodné tento soubor přejmenovat tak, aby obsahoval Váš login a (případně zkrácené) téma práce. Vyhněte se však použití mezer, diakritiky a speciálních znaků. Vhodný název může být např.: “`xlogin00-Cisteni-a-extrakce-textu.tex`”. K přejmenování můžete využít i přiložený Makefile:

```
make rename NAME=xlogin00-Cisteni-a-extrakce-textu
```

¹BP Ing. Radka Pyšného <http://www.fit.vutbr.cz/study/DP/BP.php?id=7848>

²Internetová jazyková příručka <http://prirucka.ujc.cas.cz/?id=880>

4. Vyplňte požadované položky v souboru, který byl původně pojmenován `projekt.tex`, tedy typ, rok (odevzdání), název práce, svoje jméno, ústav (dle zadání), tituly a jméno vedoucího, abstrakt, klíčová slova a další formální náležitosti.
5. Rozšířený abstrakt v češtině lze v šabloně povolit odkomentováním příslušných 2 bloků v souboru `fitthesis.cls`.
6. Nahraďte obsah souborů s kapitolami práce, literaturou a přílohami obsahem svojí technické zprávy. Jednotlivé přílohy či kapitoly práce může být výhodné uložit do samostatných souborů – rozhodnete-li se pro toto řešení, je doporučeno zachovat konvenci pro názvy souborů, přičemž za číslem bude následovat název kapitoly.
7. Nepotřebujete-li přílohy, zakomentujte příslušnou část v `projekt.tex` a příslušný soubor vyprázdněte či smažte. Nesnažte se prosím vymyslet nějakou neúčelnou přílohu jen proto, aby daný soubor bylo čím naplnit. Vhodnou přílohou může být obsah přiloženého paměťového média.
8. Nascanované zadání uložte do souboru `zadani.pdf` a povolte jeho vložení do práce parametrem šablony v `projekt.tex` (`\documentclass[zadani]{fitthesis}`).
9. Nechcete-li odkazy tisknout barevně (tedy červený obsah – bez konzultace s vedoucím nedoporučuji), budete pro tisk vytvářet druhé PDF s tím, že nastavíte parametr šablony pro tisk: (`\documentclass[zadani,print]{fitthesis}`). Barevné logo se nesmí tisknout černobíle!
10. Vzor desek, do kterých bude práce vyvázána, si vygenerujte v informačním systému fakulty u zadání. Pro disertační práci lze zapnout parametrem v šabloně (více naleznete v souboru `fitthesis.cls`).
11. Nezapomeňte, že zdrojové soubory i (obě verze) PDF musíte odevzdat na CD či jiném médiu přiloženém k technické zprávě.

Obsah práce se generuje standardním příkazem `\tableofcontents` (zahrnut v šabloně). Přílohy jsou v něm uvedeny úmyslně.

Pokyny pro oboustranný tisk

- **Oboustranný tisk je doporučeno konzultovat s vedoucím práce.**
- Je-li práce tištěna oboustranně a její tloušťka je menší než tloušťka desek, nevypadá to dobře.
- Zapíná se parametrem šablony: `\documentclass[twoside]{fitthesis}`
- Po vytištění oboustranného listu zkontrolujte, zda je při prosvícení sazební obrazec na obou stranách na stejné pozici. Méně kvalitní tiskárny s duplexní jednotkou mají často posun o 1–3 mm. Toto může být u některých tiskáren řešitelné tak, že vytisknete nejprve liché stránky, pak je dáte do stejného zásobníku a vytisknete sudé.
- Za titulním listem, obsahem, literaturou, úvodním listem příloh, seznamem příloh a případnými dalšími seznamy je třeba nechat volnou stránku, aby následující část začínala na liché stránce (`\cleardoublepage`).
- Konečný výsledek je nutné pečlivě přezkontrolovat.

Styl odstavců

Odstavce se zarovnávají do bloku a pro jejich formátování existuje více metod. U papírové literatury je častá metoda s použitím odstavcové zarážky, kdy se u jednotlivých odstavců textu odsazuje první řádek odstavce asi o jeden až dva čtverčíky (vždy o stejnou, předem zvolenou hodnotu), tedy přibližně o dvě šířky velkého písmene M základního textu. Poslední řádek předchozího odstavce a první řádek následujícího odstavce se v takovém případě neoddělují svislou mezerou. Proklad mezi těmito řádky je stejný jako proklad mezi řádky uvnitř odstavce. [2] Další metodou je odsazení odstavců, které je časté u elektronické sazby textů. První řádek odstavce se při této metodě neodsazuje a mezi odstavce se vkládá vertikální mezera o velikosti $1/2$ řádku. Obě metody lze v kvalifikační práci použít, nicméně často je vhodnější druhá z uvedených metod. Metody není vhodné kombinovat.

Jeden z výše uvedených způsobů je v šabloně nastaven jako výchozí, druhý můžete zvolit parametrem šablony “odsaz”.

Užitečné nástroje

Následující seznam není výčtem všech využitelných nástrojů. Máte-li vyzkoušený osvědčený nástroj, neváhejte jej využít. Pokud však nevíte, který nástroj si zvolit, můžete zvážit některý z následujících:

MikTeX L^AT_EX pro Windows – distribuce s jednoduchou instalací a vynikající automatizací stahování balíčků.

TeXstudio Přenositelné opensource GUI pro L^AT_EX. Ctrl+klik umožňuje přepínat mezi zdrojovým textem a PDF. Má integrovanou kontrolu pravopisu, zvýraznění syntaxe apod. Pro jeho využití je nejprve potřeba nainstalovat MikTeX.

WinEdt Ve Windows je dobrá kombinace WinEdt + MiKTeX. WinEdt je GUI pro Windows, pro jehož využití je nejprve potřeba nainstalovat **MikTeX** či **TeX Live**.

Kile Editor pro desktopové prostředí KDE (Linux). Umožňuje živé zobrazení náhledu. Pro jeho využití je potřeba mít nainstalovaný **TeX Live** a Okular.

JabRef Pěkný a jednoduchý program v Javě pro správu souborů s bibliografií (literaturou). Není potřeba se nic učit – poskytuje jednoduché okno a formulář pro editaci položek.

InkScape Přenositelný opensource editor vektorové grafiky (SVG i PDF). Vynikající nástroj pro tvorbu obrázků do odborného textu. Jeho ovládnutí je obtížnější, ale výsledky stojí za to.

GIT Vynikající pro týmovou spolupráci na projektech, ale může výrazně pomoci i jednomu autorovi. Umožňuje jednoduché verzování, zálohování a přenášení mezi více počítači.

Overleaf Online nástroj pro L^AT_EX. Přímo zobrazuje náhled a umožňuje jednoduchou spolupráci (vedoucí může průběžně sledovat psaní práce), vyhledávání ve zdrojovém textu kliknutím do PDF, kontrolu pravopisu apod. Zdarma jej však lze využít pouze s určitými omezeními (někomu stačí na disertaci, jiný na ně může narazit i při psaní bakalářské práce) a pro dlouhé texty je pomalejší.

Pozn.: Overleaf nepoužívá Makefile v šabloně – aby překlad fungoval, je nutné kliknout pravým tlačítkem na `projekt.tex` a zvolit “Set as Main File”.

Užitečné balíčky pro \LaTeX

Studenti při sazbě textu často řeší stejné problémy. Některé z nich lze vyřešit následujícími balíčky pro \LaTeX :

- `amsmath` – rozšířené možnosti sazby rovnic,
- `float`, `afterpage`, `placeins` – úprava umístění obrázků,
- `fancyvrb`, `alltt` – úpravy vlastností prostředí Verbatim,
- `makecell` – rozšíření možností tabulek,
- `pdflscape`, `rotating` – natočení stránky o 90 stupňů (pro obrázek či tabulku),
- `hyphenat` – úpravy dělení slov,
- `picture`, `epic`, `eepic` – přímé kreslení obrázků.

Některé balíčky jsou využity přímo v šabloně (v dolní části souboru `fitthesis.cls`). Nahlédnutí do jejich dokumentace může být rovněž užitečné.

Sloupec tabulky zarovnaný vlevo s pevnou šířkou je v šabloně definovaný “L” (používá se jako “p”).