

Studijní materiály z matematiky ve formátu HTML

Mathematical Study Materials in HTML Format

Marcel Machala

Bakalářská práce
2012



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2011/2012

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Marcel MACHALA**
Osobní číslo: **A09058**
Studijní program: **B 3902 Inženýrská informatika**
Studijní obor: **Informační a řídicí technologie**

Téma práce: **Studijní materiál z matematiky ve formátu HTML**

Zásady pro vypracování:

1. Sestavte přehled a zhodnoťte kvalitu výukových materiálů z matematiky, které jsou dostupné pro studenty na jiných VŠ ve formátu HTML.
2. Popište práci se softwarem Mathematica při vytváření HTML dokumentů.
3. Nastudujte základní principy psaní dokumentů v LaTeXu.
4. Popište možnosti LaTeXu pro vytváření HTML dokumentů (nastudujte TeX4ht, LaTeX2HTML, LaTeXML, aj.).
5. Vyberte vhodný nástroj LaTeXu a srovnajte kvalitu výsledků a náročnost tvorby HTML s ostatními programy (editory) pro tvorbu HTML.
6. Vytvořte ukázkou učebního textu z kapitoly Určitý integrál z elektronických skript Ostravský, Polášek: Diferenciální a integrální počet funkce jedné proměnné ve formátu HTML.

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. OSTRAVSKÝ, Jan a Vladimír POLÁŠEK. Diferenciální a integrální počet funkce jedné proměnné. Zlín: Univerzita Tomáše Bati ve Zlíně, 2011. ISBN 978-80-7454-124-7. Dostupné z: https://web.fai.utb.cz/cs/docs/Skripta_Matematika.1_2011.zip
2. KOPKA, Helmut a Patrick W DALY. Latex: podrobný průvodce. Vyd. 1. Překlad Jan Gregor. Brno: Computer Press, 2004, 576 s. ISBN 80-722-6973-9.
3. RYBIČKA, Jiří. Latex pro začátečníky. 3., přeprac. vyd. Brno: Konvoj, 2003, 238 s. ISBN 80-730-2049-1.
4. TEAGUE, Jason Cranford. DHTML a CSS pro World Wide Web: praktická vizuální příručka. Př. Jan GREGOR. Praha: SoftPress, 2005, 522 s. ISBN 80-864-9777-1.
5. CASTRO, Elizabeth. HTML, XHTML a CSS: názorný průvodce tvorbou WWW stránek. Vyd. 1. Brno: Computer Press, 2007, 438 s. ISBN 978-802-5115-312.
6. The Mathematica book. 5. ed. Champaign, Ill: Wolfram Media, 2003. ISBN 15-795-5022-3.

Vedoucí bakalářské práce:

RNDr. Jan Ostravský, CSc.

Ústav matematiky

Datum zadání bakalářské práce:

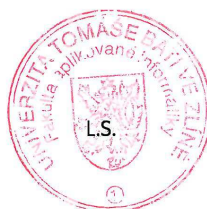
24. února 2012

Termín odevzdání bakalářské práce:

8. června 2012

Ve Zlíně dne 24. února 2012

prof. Ing. Vladimír Vašek, CSc.
děkan



prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

ABSTRAKT

Práce se zabývá prostředky tvorby a tvorbou výukových materiálů do předmětu Matematika 1 na Fakultě aplikované informatiky UTB, zejména zobrazováním matematických vzorců. Práce obsahuje srovnání materiálů několika vysokých škol, popis práce v systémech LaTeX a Mathematica, dále základy tvorby webových stránek. V praktické části popisuje tvorbu webových stránek se skripty výše uvedeného předmětu.

Klíčová slova: HTML, matematika, skripta, Mathematica, LaTeX, jsMath, MathML

ABSTRACT

The thesis focuses on the means of production and on producing educational materials for Mathematics 1 at the Faculty of Applied Informatics UTB, mainly by portraying mathematical formulas. The thesis is comparing materials from various universities, description of work with systems LaTeX and Mathematica and basics of website production. In the practical part the thesis is depicting the production of websites containing lecture notes.

Keywords: HTML, textbook, mathematics, Mathematica, LaTeX, jsMath, MathML

Tímto bych chtěl poděkovat vedoucímu mé bakalářské práce panu RNDr. Janu Ostravskému, CSc. za cenné rady a připomínky, zejména v hektickém závěru.

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....

podpis diplomanta

OBSAH

ÚVOD	10
1 TEORETICKÁ ČÁST	11
1 STUDIJNÍ MATERIÁLY OSTATNÍCH VYSOKÝCH ŠKOL	12
1.1 VYSOKÁ ŠKOLA BÁŇSKÁ - TECHNICKÁ UNIVERZITA OSTRAVA.....	12
1.2 VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ.....	13
1.3 ZÁPADOČESKÁ UNIVERZITA V PLZNI	14
2 LATEX	16
2.1 ZDROJOVÝ TEXT	17
2.1.1 Hlavička (preamble).....	18
2.1.2 Tělo dokumentu	19
2.2 REŽIMY ZPRACOVÁNÍ TEXTU V LATEXU.....	19
2.3 VYTVOŘENÍ LATEXOVÉHO DOKUMENTU	20
2.4 PŘÍKAZY LATEX	20
2.5 SAZBA TEXTU	21
2.5.1 Změna písma	21
2.5.1.1 Zvýrazňování	21
2.5.1.2 Velikost písma	21
2.5.2 Centrování	22
2.5.2.1 Centrování na střed	22
2.5.2.2 Sazba na praporek	22
2.5.2.3 Zúžená sazba.....	22
2.5.2.4 Seznamy.....	22
2.5.3 Boxy	23
2.5.3.1 LR box	23
2.5.3.2 Parbox a minipage	23
2.5.3.3 Rule box.....	23
2.5.4 Tabulky.....	24
2.6 MATEMATICKÉ VZORCE.....	24
2.6.1 Horní a dolní indexy.....	24
2.6.2 Zlomky	25
2.6.3 Odmocniny	25
2.6.4 Sumy a integrály.....	25
2.7 MATEMATICKÉ SYMBOLY	25
2.8 GRAFIKA	25
2.8.1 Vlastní grafické možnosti LaTeXu	25
2.8.2 Import externí grafiky.....	26
3 MATHEMATICA	27

3.1	JAK MATHEMATICA PRACUJE	28
3.2	PŘÍKAZY	29
3.2.1	Aritmetické operace	29
3.2.2	Základní funkce	29
3.2.3	Proměnné	30
3.2.4	Funkce	30
3.2.5	Komplexní čísla	30
3.2.6	Vektory, matice	31
3.2.7	Funkce Table	31
3.2.8	Tabulky	31
3.2.9	Grafy funkcí	32
3.2.9.1	Funkce Plot	32
3.2.9.2	Funkce Plot3D	32
3.2.9.3	Funkce Show	33
3.2.9.4	Ostatní typy grafů	33
3.2.10	Derivace	33
3.2.11	Integrál	33
3.3	TEXT	33
3.4	NÁPOVĚDA	33
4	TVORBA DOKUMENTŮ VE FORMÁTU HTML	35
4.1	VYTVOŘENÍ ZDROJOVÉHO KÓDU	35
4.1.1	Hlavička	36
4.1.2	Tělo (body) dokumentu	37
4.2	ŘÁDKOVÉ ZNAČKY HTML	37
4.2.1	Fyzické formátování HTML	37
4.2.2	Logické formátování HTML	38
4.3	BLOKOVÉ ZNAČKY HTML	38
4.4	SEZNAMY	39
4.4.1	Číslovaný seznam	39
4.4.2	Odrážkový seznam	39
4.4.3	Vnořené seznamy	39
4.4.4	Další typy seznamů	39
4.5	ODKAZY	40
4.6	TABULKY	40
II	PRAKTICKÁ ČÁST	41
5	DOKUMENTY VE FORMÁTU HTML OBSAHUJÍCÍ MATEMATICKÉ VZORCE	42
5.1	VKLÁDÁNÍ VZORCŮ JAKO OBRÁZKY	43
5.2	VKLÁDÁNÍ VZORCŮ POMOCÍ JAVASCRIPTU	45
5.2.1	Příprava	45
5.2.2	Vlastní zápis vzorců	45
5.2.3	Vlastnosti stránky s použitím jsMath	46

5.3	VKLÁDÁNÍ VZORCŮ POMOCÍ JAZYKA MATHML	47
6	EXPORT Z LATEXU	49
6.1	LATEX2HTML	49
6.2	TeX4HT	49
6.3	LATEXML	50
7	EXPORT ZE SOFTWARE MATHEMATICA	51
8	POROVNÁNÍ KVALITY EXPORTU A NÁROČNOST TVORBY.....	55
9	TVORBA VYBRANÉ KAPITOLY V HTML.....	56
9.1	KOMPLEXNÍ VZHLED STRÁNEK	56
9.2	TVORBA SAMOTNÝCH SKRIPT	56
ZÁVĚR	58	
ZÁVĚR V ANGLIČTINĚ.....	59	
SEZNAM POUŽITÉ LITERATURY.....	60	
SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK	62	
SEZNAM OBRÁZKŮ	63	
SEZNAM PŘÍLOH.....	64	

ÚVOD

U každého studia jsou důležité studijní materiály. Je jedno jedná-li se o samostudium, studium na vysoké škole nebo třeba studium s lektorem. Pro každý typ studia je některá forma materiálů více nebo méně vhodná než jiná. Ale čím kvalitnější materiály jsou, tím je větší šance, že studenta zaujmou, a že jeho studium bude úspěšné.

Kvalitní studijní materiály by měly obsahovat více než strohé věty, definice a vzorce, jež je potřeba se naučit. Měly by obsahovat i „omáčku“ kolem, aby student mohl vidět a pochopit širší souvislosti, použití v praxi apod.

Moderní studijní materiály v dnešní době jsou v každém případě elektronické. Tyto jsou doby, kdy studenti všude nosili spoustu těžkých knih. Snad úplně každý student má přístup k PC, kde může elektronické materiály studovat. Navíc velké procento studentů vlastní notebook nebo tzv. chytrý telefon s přístupem na Internet, proto se nabízí vystavení elektronických materiálů na web.

A kvalitní moderní studijní materiály jsou logicky sloučením dvou předchozích skupin. Měly by být online, nejlépe interaktivní se spoustou příkladů, s rozvedením tématu a s praktickými ukázkami (reálné video, animace apod.).

V teoretické části se budu věnovat studijním materiálům z matematiky ostatních vysokých škol a jejich prezentaci na internetu, základnímu popisu programů LaTeX a Mathematica a základům tvorby dokumentů ve formátu HTML.

Praktická část se bude zabývat tvorbou dokumentu ve formátu HTML obsahujícím matematické vzorce klasickou cestou (HTML editor), exportem z LaTeXu a exportem ze softwaru Mathematica. V této části bude popsán i postup primárního cíle této práce – tvorba materiálu ve formátu HTML k vybrané kapitole z elektronických skript.

I. TEORETICKÁ ČÁST

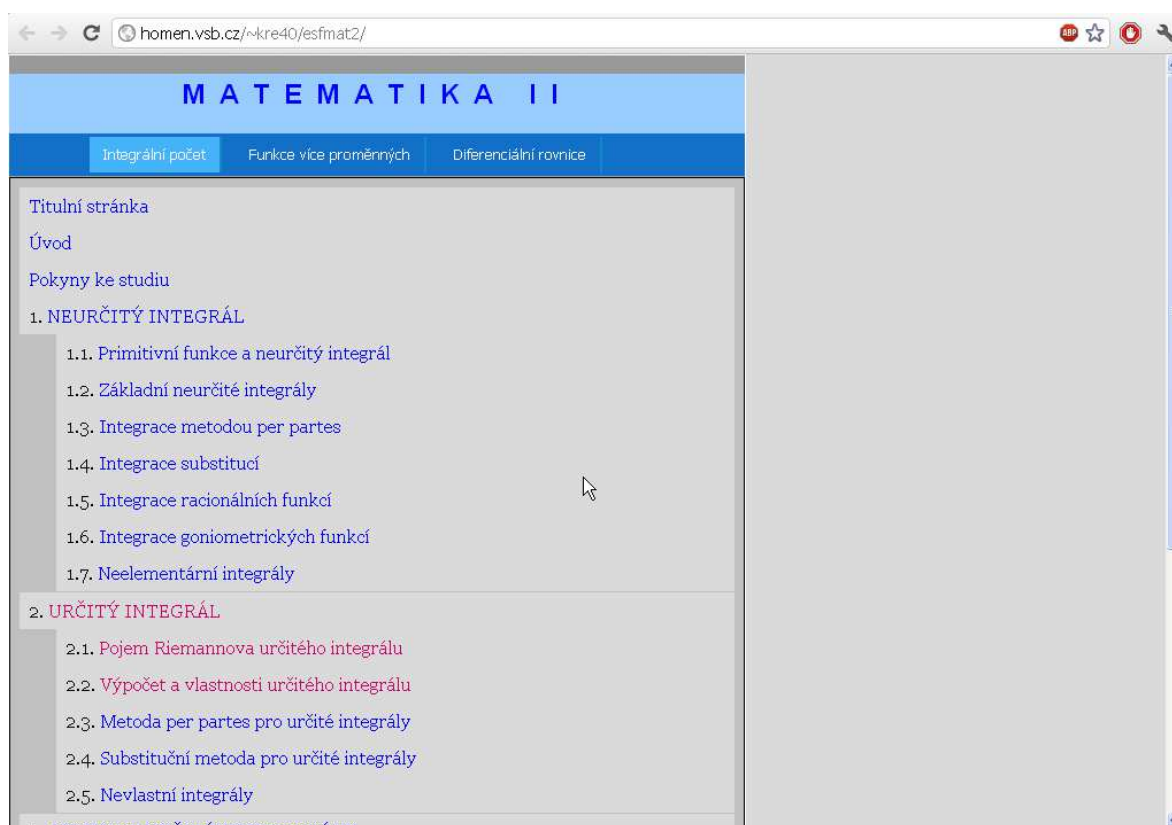
1 STUDIJNÍ MATERIÁLY OSTATNÍCH VYSOKÝCH ŠKOL

Pro porovnání jsem vybral tři vysoké školy (Vysoká škola báňská – Technická univerzita Ostrava, Vysoké učení technické v Brně a Západočeská univerzita v Plzni). Každá z nich má trochu jiný přístup k zveřejňování studijních materiálů.

1.1 Vysoká škola báňská - Technická univerzita Ostrava

Materiály jsou uloženy na webu [STUDIJNÍ OPORY S PŘEVAŽUJÍCÍMI DISTANČNÍMI PRVKY PRO PŘEDMĚTY TEORETICKÉHO ZÁKLADU STUDIA](http://www.studopory.vsb.cz) [www.studopory.vsb.cz] v sekci Studijní materiály.

Materiály mají formát *.pdf. Každá kapitola je uložena jako jeden soubor. Nejsou nijak provázané, pro získání nové kapitoly se musíme vrátit na hlavní stránku předmětu.



Obr. 1. Stránka materiálů VŠB-TU Ostrava

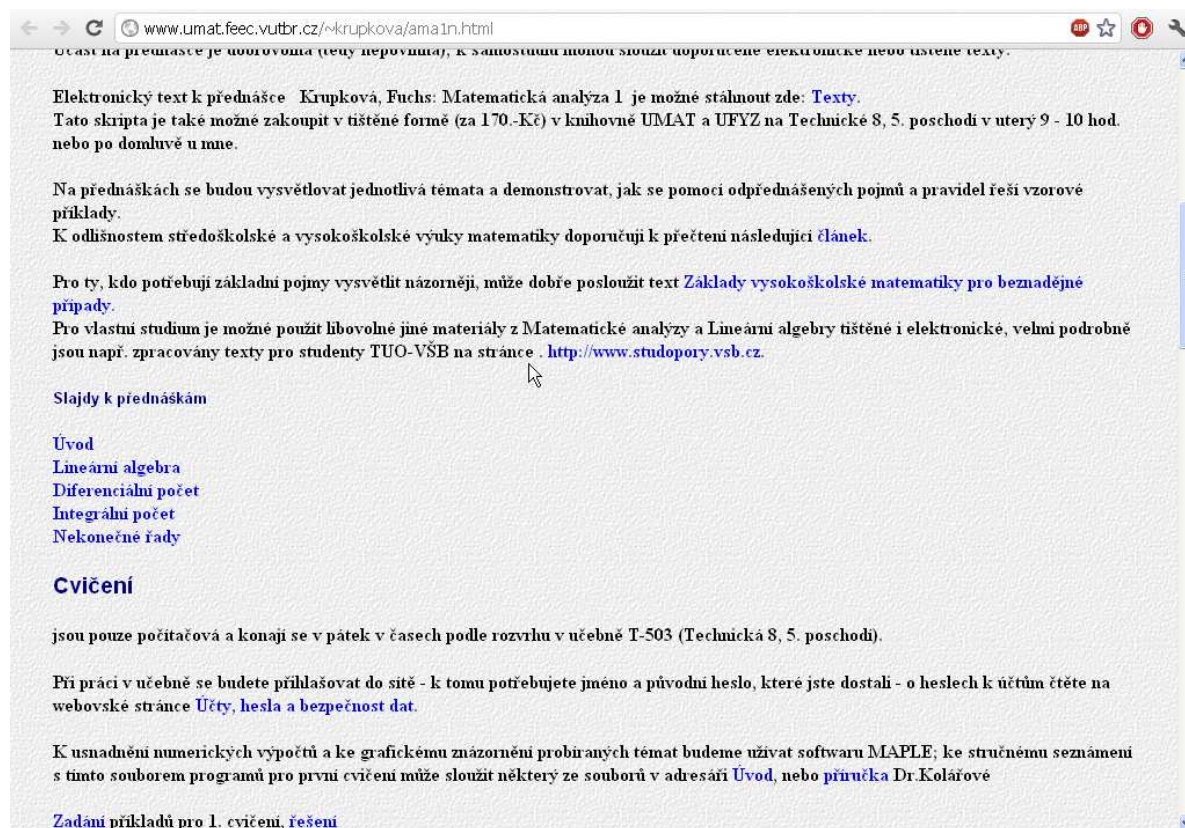
Vytvoření a správa takového webu není nijak obtížná. Stránky jsou přehledné, ale působí takovým „suchým“ dojmem. I když dnešní internetové prohlížeče zobrazují *.pdf soubory pomocí plug-inů přímo ve svém hlavním okně, tzn. soubor *.pdf se zobrazuje jako webová stránka, neprovázanost jednotlivých kapitol pomocí odkazů nutí návštěvníka stránek k neustálému vracení se na úvodní stránku předmětu.

1.2 Vysoké učení technické v Brně

Materiály má každý vyučující uložené pod svým profilem na webových stránkách fakulty. Vybral jsem stránku s materiály pro výuku předmětu [MATEMATIKA 1 \(AMA\) PANÍ RNDR. VLASTY KRUPKOVÉ, CSc.](http://www.umat.feec.vutbr.cz/~krupkova/ama1n.html) [<http://www.umat.feec.vutbr.cz/~krupkova/ama1n.html>].

Jsou zde uložena kompletní skripta k předmětu ve formátu **.pdf*; dále zde můžeme najít přednáškové slajdy ve formátu **.ppt* (prezentace PowerPoint) a totéž ve formátu **.pdf*.

Nakonec najdeme několik souborů s příklady, ve kterých je nutné používat program Maple, a řešení příkladů. Pro zobrazení nebo stažení další části se musíme opět vrátit na hlavní stránku.



Obr. 2. Stránka materiálů VUT v Brně

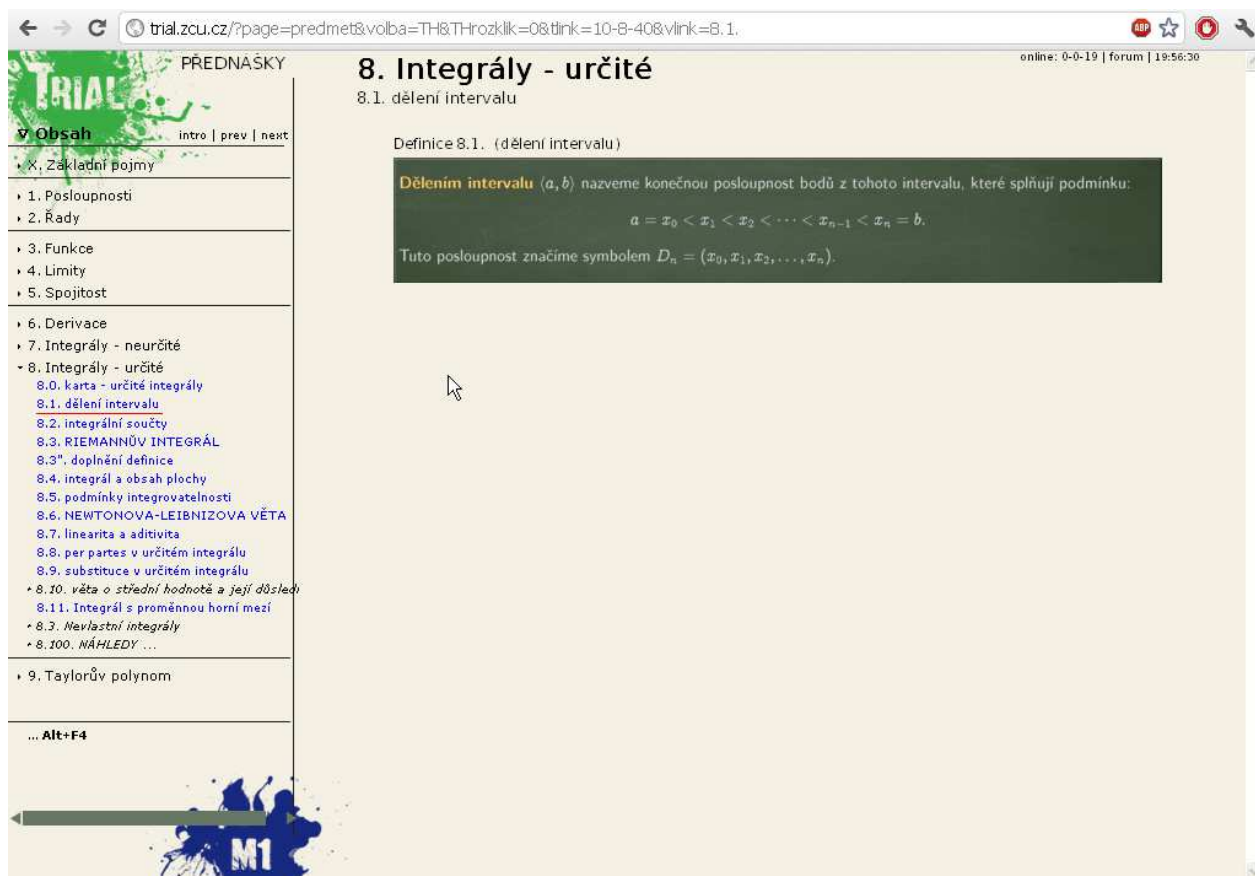
VUT v Brně nemá centrální úložiště studijních materiálů, ale každý vyučující má svoje materiály uložené ve svém diskovém prostoru. Pokud více vyučujících používá stejné materiály, mohou použít hypertextových odkazů k odkazování, aby nemusely být uloženy duplicitně. Pokud má vyučující několik skupin studentů, kteří mají odlišné materiály, mohou být stránky poněkud nepřehledné.

I když některé internetové prohlížeče dokážou zobrazovat prezentace PowerPointu přímo ve svém hlavním okně, působí stránky dost staticky a většina návštěvníků si soubory stáhne do počítače, aby je explicitní aplikací spustila.

1.3 Západočeská univerzita v Plzni

Na webu [TRIAL](http://trial.zcu.cz) [trial.zcu.cz] najdeme soupis definic a vět, a to přímo ve formátu HTML a zároveň i ke stažení ve formátu *.pdf.

V části věnované příkladům je množství příkladů ke všem okruhům témat i s postupem řešení. Řešení je skryté a zobrazí se teprve po kliknutí, takže se student může pokusit o samostatné řešení, které pak zkonfrontuje s řešením na internetu.



Obr. 3. Stránka materiálů ZČU v Plzni

Stránky ZČU v Plzni jsou příkladem moderního webu, který se snaží být interaktivní a zapojit aktivně návštěvníka stránky. Je škoda, že není více rozebírána teorie, protože samotné věty a vzorce často nedokážou vysvětlit princip a důvod výpočtu.

Všechny stránky jsou provázané pomocí menu na levé straně, kde se návštěvník dostane téměř kamkoliv na dvě kliknutí. Vzorce se zobrazují pomocí obrázků ve formátu *.png*.

2 LATEX

LaTeX je nadstavba základního systému TeX, který naprogramoval v roce 1984 Donald E. Knuth, který nebyl spokojený se sazbou matematických vzorců v jeho knize. Je oblíbený zvláště ve vědeckých kruzích pro vysokou úroveň typologické sazby a vysokou přesnost při sázení dokumentů - zejména matematických. [3]

TeX je zkratkou řeckého slova „technika“ a čte se „tech“ nebo „tek“ (z angličtiny).

TeX se podobá programovacímu jazyku, jehož příkazy („primitivní funkce“) se zapisují do zdrojového textu, typicky uvozené zpětným lomítkem.

Protože psaní textu pro zpracování TeXem je pro začátečníky obtížné a složité začaly vznikat nadstavby TeXu. Nejznámější z volně šiřitelných je LaTeX. Je to sestava maker, které usnadňují zadávání příkazů. LaTeX sestavil v roce 1985 Leslie Lamport a je od té doby neustále vyvíjen a rozšiřován. Neřízeným rozšiřováním se LaTeX stával postupně zpětně nekompatibilním, proto byl v roce 1992 sestaven standart LaTeX 2.09; jeho revize se označují pouze datem vzniku. V současné době je používán standart LaTeX 2 ϵ a připravuje se nový standart LaTeX 3.[2,3]

Díky LaTeXu může vytvářet i uživatel, který nemá žádné sazečské znalosti, dokumenty na vysoké úrovni. LaTeX

Práce se systémem TeX a jeho nadstavbami se diametrálně liší od systémů s grafickým uživatelským rozhraním. Zejména pro začátečníky je velmi odrazující, že při práci není k dispozici okamžitý náhled na výsledný dokument. Málokdo si ale uvědomí, že úpravy ve výsledném tvaru textu jsou namáhavé a předpokládají únavnou a monotónní práci s myší. Systém TeX očekává zápis několika příkazů a sám se postará o precizní a bezchybné vysázení textu.

Mezi další výhody patří:

- Možnost použití libovolného textového editoru. Uživatel může používat program, na který je zvyklý.
- Možnost použití různých filtrů a programů na automatickou úpravu zdrojového textu. Například převod různých kódování, jazykovou korekci, apod.
- Využití všech možností operačního systému ke zdokonalení funkce systému. Překlad se spouští z příkazového řádku, můžeme vytvořit příkazové dávky (skripty)

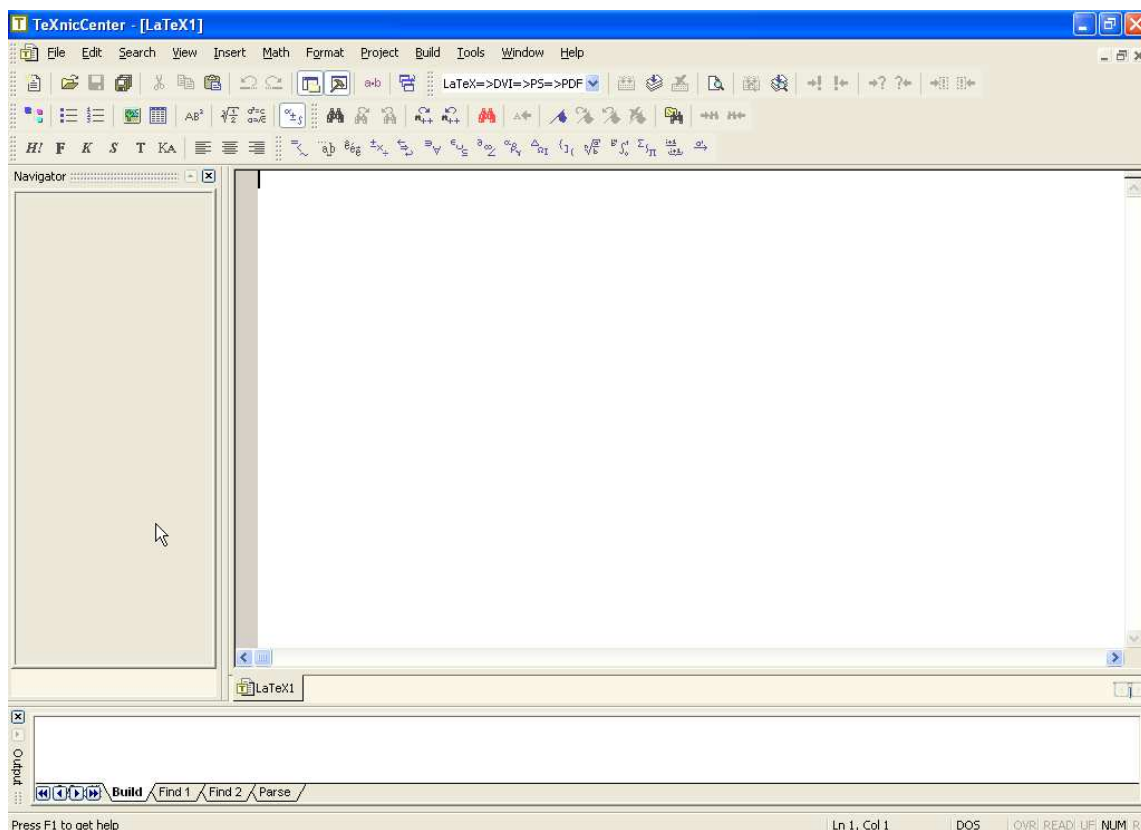
usnadňující rutinní práci. Můžeme použít systémové textové filtry (např. grep), pro extrakci informací ze zdrojového textu.

- Snadná přenositelnost, bezpečnost a archivace zdrojových textů. Vždy se jedná o soubor ASCII znaků, který je snadno čitelný, i když je částečně poškozený.[2,3]

2.1 Zdrojový text

Zdrojový text obsahuje kromě samotného textu, který má být vtištěn, i tzv. příkazy. Příkazem může být určitý jednotlivý znak nebo slovo uvozené speciálním znakem – zpětným lomítkem.[3]

Zdrojový text je možné napsat v jakémkoliv textovém editoru, který umí uložit soubor do obvyčejného ASCII kódu (např. notepad). Ale existují i specializované editory (např. Texmaker, TeXnicCenter a další), které usnadňují zadávání příkazů.



Obr. 4. Editor TeXnicCenter

Některé příkazy mohou být doplněny o argumenty (parametry). Parametry se zapisují do složených, hranatých nebo oblých závorek v závislosti na typu příkazu a druhu parametru.

Zdrojový text pro zpracování LaTeXem se dělí na dvě části: hlavičku a samotné tělo dokumentu.

2.1.1 Hlavička (preamble)

V hlavičce se zapisuje formátování platné pro celý dokument (globální formátování). Začíná povinným úvodním příkazem `\documentclass`, který má syntax `\documentclass[volby]{třída}[datum vytvoření]`. Parametr třída je povinný a definuje základní styl sazby. Zároveň je to jméno souboru s příponou `*.cls`, v němž je definice třídy uvedena. Standartní třídy jsou: *article* (článek), *report* (zpráva), *book* (kniha), *letter* (dopis).

Soubory `*.cls` jsou obyčejné textové ASCII soubory s definicemi příkazů, které lze libovolně upravovat nebo vytvářet vlastní.

Příkaz `\documentclass` má nepovinný parametr volby, kterým lze modifikovat činnost příkazů ve zvolené třídě. Například:

- `11pt` – sazba celého dokumentu bude provedena tak, že základní velikost písma bude 11pt a všechny ostatní velikosti budou úměrně přizpůsobeny
- `twoside` – rozlišování levých a pravých stránek (liché stránky jsou v knize na pravé straně) – je upraveno číslování stránek a některé další vlastnosti
- `a4paper` – nastavení formátu stránky na A4. Podobně existují i volby `a5paper`, `b5paper`, `letterpaper`, `legalpaper`
- `landscape` – výstup bude formátován „na šířku“, tj. rozměry výšky a šířky stránky budou zaměněny

V parametru lze uvést několik voleb, které musí být odděleny čárkami bez mezer, například: `\documentclass[a4paper,12pt,twoside]{book}`

Dále lze v hlavičce připojovat další balíky příkazů – *packages*, jejichž příkazy mohou být v dokumentu použity. Připojením se provádí příkazem `\usepackage`, který má stejnou syntax jako příkaz `\documentclass`, tj. `\usepackage [volby]{balík}[datum vytvoření]`. Můžeme použít několik příkazů `\usepackage` za sebou. Hlavička dokumentu pro LaTeX může vypadat například takto:

```
\documentclass[a4paper,onesize]{article}
```

%papír A4, bez rozlišení lichých a sudých stránek, třída je článek

```
\usepackage[czech]{babel} %jazyková mutace
```

```
\usepackage[cp1250]{inputenc} %kódování znaků pro Windows
```

Vše co je za znakem % překladač až do konce řádku ignoruje – poznámka.[2]

2.1.2 Tělo dokumentu

V těle dokumentu je samotný text, který chceme vysázet, spolu s formátovacími značkami a příkazy, kterými se vkládají tabulky, obrázky, atd. Tělo dokumentu začíná příkazem `\begin{document}` a končí příkazem `\end{document}`.

Část zdrojového textu mezi příkazy `\begin{název}` a `\end{název}` se nazývá prostředí. Prostedí dělí dokument na menší logické celky, které spolu souvisí. Prostedí se můžou libovolně vnořovat, nesmějí se křížit. Některé příkazy ovlivňují sazbu následujícího textu až do konce prostředí, v němž byly uvedeny. Hranicemi tedy nastavujeme úsek vlivu těchto příkazů.

Největší prostředí dokumentu je tedy *document*. [2]

2.2 Režimy zpracování textu v LaTeXu

Existují tři režimy zpracování textu:

- Odstavcový režim
- Matematický režim
- Režim LR (left-right)

Odstavcový režim je běžný režim, kdy LaTeX zpracovává vstupní text jako posloupnost slov do řádků, odstavců a stránek.

Matematický režim je režim, do kterého se LaTeX přepne, narazí-li na příkaz, který označuje vzorec. V tomto režimu LaTeX ignoruje mezery. Po příkazu, který ukončuje vzorec, se LaTeX přepne zpět do odstavcového režimu.

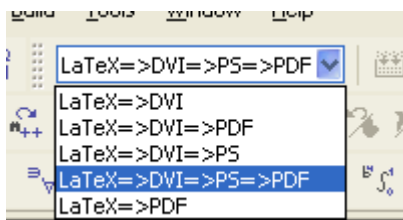
Režim LR je speciální režim, kdy LaTeX zpracovává vstupní text jako jeden řádek, bez možnosti zalomení řádku. Jedná se například o vložený text do vzorce nebo argument příkazu `\mbox{text v režimu LR}`. [2]

2.3 Vytvoření LaTeXového dokumentu

Vytvoření dokumentu určeného pro tisk má tři kroky:

- Vytvoření (úprava) zdrojového textu (vlastní text dokumentu + příkazy LaTeXu pro formátování), většinou s příponou **.tex*
- Zpracování zdrojového textu LaTeXem. Většina distribucí má samostatný program **latex**, jako argument následuje jméno souboru se zdrojovým textem. Jako výstup se vytvoří soubor se stejným jménem, a příponou **.dvi*. Tento soubor obsahuje zformátovaný text, který je ale nezávislý výstupním zařízením.
- Soubor **.dvi* se zkonvertuje do formátu zvolené tiskárny, nejčastěji do formátu *PostScript*, který podporuje většina tiskáren, popř. do formátu **.pdf*, který se zobrazí na všech platformách stejně.

Některé specializované editory pro práci s LaTeXem dělají všechny tři kroky automaticky.



Obr. 5. Možnost exportu z LaTeXu v editoru TeXnicCenter

2.4 Příkazy LaTeX

Existují tři typy příkazů:

- Samostatné znaky # \$ & ~ _ ^ % { }
- Zpětné lomítko \ plus řídicí znak (např. \ \$ vytiskne symbol \$)
- Zpětné lomítko \ plus posloupnost písmen (např. \ large přepne na větší typ písma)

Většina příkazů vyžaduje zadání argumentu, který se zapisuje do složených závorek. Takové argumenty nazýváme povinné (např. \emph{FAI} vytiskne *FAI*). Dalším typem argumentů jsou volitelné argumenty. Volitelné argumenty se zapisují do hranatých závorek (např. \rule [3mm] {5mm} {2mm} vytiskne černý obdélník široký 5mm a vysoký 2mm 3mm nad účaří).[3]

2.5 Sazba textu

Text může vysadit a zvýraznit např. změnou typu a řezu nebo velikosti písma, centrováním, odsazením, vyznačením odstavců, atd.

2.5.1 Změna písma

Písmem neboli fontem nazýváme množinu písmen, číslic a znaků určité velikost a vzhledu. Základním fontem je antikva střední šířky o velikosti specifikované v hlavičce dokumentu. Vybíráme z velikostí 10, 11 nebo 12 bodů.

2.5.1.1 Zvýrazňování

Nejčastěji se používá zvýraznění kurzívou. Přepnutí na kurzívu se v LaTeXu provede příkazem `\emph{text_ke_zvýraznění}`.

2.5.1.2 Velikost písma

V LaTeXu máme tyto možnosti volby velikosti písma:

- `\tiny` nejmenší
- `\scriptsize` velmi malé
- `\footnotesize` menší
- `\small` malé
- `\normalsize` normální
- `\large` velké
- `\Large` větší
- `\LARGE` ještě větší
- `\huge` velmi velké
- `\Huge` největší

Velikost je relativní ke standartní velikosti zvolené v hlavičce dokumentu. Přepínače změny velikosti písma buď do dalšího přepínače, nebo do konce prostředí.

2.5.2 Centrování

2.5.2.1 Centrování na střed

Využívá se prostředí `{center}`. V tomto prostředí se centruje text mezi příkazy `\`. Pokud se text nevejde na jeden řádek, rozdělí se na více řádků pomocí rovnoměrných mezislovních mezer. Nedochozí k dělení slov.

2.5.2.2 Sazba na praporek

Využívá se prostředí `{flushleft}` (zarovnání textu vlevo) nebo `{flushright}` (zarovnání textu vpravo). Zarovnání probíhá mezi příkazy `\`. Pokud se text nevejde na jeden řádek, rozdělí se na více řádků pomocí rovnoměrných mezislovních mezer. Nedochozí k dělení slov.

2.5.2.3 Zúžená sazba

Pro sazbu užšího bloku textu má LaTeX dvě prostředí:

- `{quote}` – větší mezera mezi odstavci
- `{quotation}` – odsazení prvního řádku odstavce

2.5.2.4 Seznamy

K vytváření seznamů můžeme použít tři prostředí:

- `{itemize}`
- `{enumerate}`
- `{description}`

Jednotlivé položky seznamu se uvozují příkazem `\item`. V prostředí **itemize** jsou položky označeny černým puntíkem – bullet. V prostředí **enumerate** jsou návěštím pořadová čísla. Při každém zavolání prostředí se začíná číslovat od 1. V prostředí **description** má příkaz `\item` volitelný parametr, který vystupuje jako návěští.

Seznamy můžeme libovolně vnořovat až do čtyř úrovní vnoření. Každá úroveň je o něco více odsazená. Každá úroveň zanoření má jiný typ návěští.

2.5.3 Boxy

Box je část dokumentu, se kterou TeX pracuje jako by to byl jeden znak o velikosti dané jeho obsahem. Lze ho přemísťovat, ale nelze ho dělit, ale lze je spojovat. V LaTeXu jsou implementovány tři druhy boxů:

- LR box
- Odstavcový box
- Linkový (rule) box

2.5.3.1 LR box

LR box obsahuje materiál v jednom řádku zleva doprava. LR box můžeme vytvořit následujícími příkazy:

- `\mbox{text}` – vytvoří box o šířce `text`
- `\fbox{text}` – vytvoří orámovaný box o šířce `text`
- `\makebox[šířka][poz]{text}` – vytvoří box o šířce `šířka`, `poz` specifikuje umístění textu v boxu
- `\framebox[šířka][poz]{text}` – vytvoří orámovaný box o šířce `šířka`, `poz` specifikuje umístění textu v boxu

2.5.3.2 Parbox a minipage

Celé odstavce můžeme vložit do vertikálních (odstavcových) boxů příkazem `\parbox[poz]{šířka}{text}` nebo můžeme použít prostředí pro malé stránky `{minipage}`. Obojí vytvoří svislý box o šířce `šířka`, kde je text v klasickém odstavcovém režimu.

2.5.3.3 Rule box

Rule box je černý obdélník. Syntaxe je `\rule[zvednutí]{šířka}{výška}`. Toto vytvoří černý obdélník o šířce `šířka` a výšce `výška`, zvednutý na účaří `zvednutí`.

Je možné vytvořit rule box o nulové šířce ale nenulové výšce, tím můžeme zvětšit výšku boxu na libovolnou hodnotu.

2.5.4 Tabulky

I když je možné pomocí boxů a prostředí `{tabbing}` vytvořit všechny typy tabulek, existuje mnohem pohodlnější metoda k vytváření tabulek.

Je to prostředí `{tabular}` se syntaxí `\begin{tabular}[poz]{sloupce}řádky\end{tabular}`. **Poz** definuje svislé umístění v boxu. **Sloupce** definuje formátování sloupců. **Řádky** představují skutečné záznamy v tabulce. Jednotlivé sloupcové záznamy se oddělují znakem `&` a každý vodorovný řádek je ukončen příkazem `\\`. [2]

2.6 Matematické vzorce

Matematické vzorce byly původní důvod vzniku systému TeX. Proto nikoho neudiví, že zápis matematických vzorců je dotažen k dokonalosti. Zápis matematických vzorců probíhá v matematickém prostředí.

Matematické vzorce mohou být dvojího druhu, textové vzorce anebo vysazené vzorce. Textové vzorce jsou na řádku textu např. $(a+b)^2=a^2+2ab+b^2$. Vysazené vzorce se používají zejména při použití velkých znaků jako je suma \sum nebo integrál \int , nebo pokud chceme vzorec zvýraznit a umístit do samostatného bloku.

Textové vzorce zapisujeme pomocí tvaru `\(text_vzorce \)` nebo `$text_vzorce$`. Vysazené vzorce zapíšeme pomocí tvaru `\[text_vzorce \]` nebo `$$text_vzorce$$`.

2.6.1 Horní a dolní indexy

Vzorce obsahují často exponenty a indexy. I když mají matematicky odlišný význam, z hlediska typologie jde jen o horní a dolní indexy. LaTeX i TeX umožňují stvořit jakoukoliv kombinaci exponentů a indexu se správnou velikostí písma. Znak `^` nastaví následující znak jako exponent, a znak `_` nastaví následující znak jako index.

Obsahuje-li exponent nebo index více než jeden znak, musí se tyto znaky umístit do skupiny, tj. do složených závorek:

`x^{y_4}`

2.6.2 Zlomky

Krátké zlomky se nejlépe prezentují pomocí znaku „ / “. Pokud chceme zapsat čitatele nad jmenovatel, musíme použít příkaz `\frac{čitatele}{jmenovatel}`. Zlomky můžeme vnořovat do jakékoliv úrovně. LaTeX ovšem sází vnořené zlomky implicitně menším řezem, pokud nenastavíme jinak.

2.6.3 Odmocniny

Odmocnina se sází pomocí příkazu:

```
\sqrt[n]{arg}
```

Pokud není zadáný volitelný argument **n**, generuje se druhá odmocnina. Výška a délka odmocniny se automaticky přizpůsobuje zadanému argumentu. Odmocniny lze libovolně vnořovat.

2.6.4 Sumy a integrály

Suma se sází příkazem `\sum`, integrál příkazem `\int`. Jak sumy, tak integrály mají často horní a dolní meze. Umístění závisí na tom, jedná-li se o vzorec textový nebo vysazený. Meze se zapisují stejně jako mocniny a indexy.

2.7 Matematické symboly

Používaných matematických symbolů jsou stovky a nemá smysl je všechny vypisovat, dají se najít v referenčních příručkách nebo na internetu. K nejpoužívanějším patří řecké písmena, které se zadávají pomocí `\` - zpětného lomítka a slovního popisu písmene (např. `\alpha` nebo `\pi`). Dále se jedná o kaligrafická písmena, binární operátory, relační operátory, šipky a ukazatele.

Do této skupiny řadíme i názvy funkcí, např. sinus => `\sin`, limita => `\lim`, apod.

2.8 Grafika

2.8.1 Vlastní grafické možnosti LaTeXu

Standartní LaTeX obsahuje prostředky k vytváření primitivní grafiky.

Obrázky se vytváří v prostředí `picture` s parametry **rozměr_x** a **rozměr_y**, které se zadávají do kulatých závorek. Základní elementy se vkládají do obrázku pomocí příkazů `\put` a `\multiput`. Vkládat se dá text, rámečky, úsečky, vektory, kružnice a kruhy, ovály a křivky.

2.8.2 Import externí grafiky

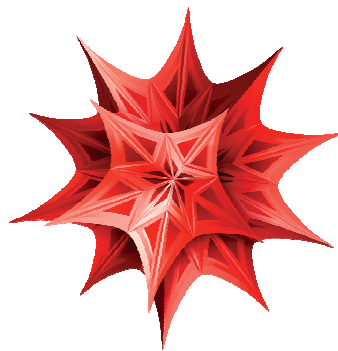
Pro import a manipulaci s externími grafickými soubory existují dva balíčky: `graphics` a `graphicx`. Liší se pouze v syntaxi příkazů.

Import externího obrázku provedeme příkazem `\includegraphics [llx, lly] [urx, ury]{název_souboru}`, kde `llx` a `lly` jsou souřadnice spodního levého rohu a `urx`, `ury` jsou souřadnice pravého horního rohu. [2]

3 MATHEMATICA

Software Wolfram Mathematica je komplexní výpočtový program, který má za sebou více než 20 let vývoje. V roce 1987 založil Stephen Wolfram společnost Wolfram Research, kterou dodnes vede. Již v roce 1988 vyšla první verze software **Mathematica 1.0**, která ukázala vizi, spojit do té doby samostatné aplikace (pro numerické výpočty, pro symbolické výpočty, pro grafické výstupy, a další) do jediného systému, který zvládne všechny aspekty různých druhů výpočtů.

Toho bylo dosaženo vyvinutím nového symbolického jazyka, který umí manipulovat se širokým spektrem objektů, ale používá málo základních primitiv. Ukázkovým případem může být $f(x)$. Tento výraz může být funkcí, grafikou, zvukem, programem i celým dokumentem.[4]



Obr. 6. Logo Wolfram Mathematica 8 (Spikey)

Wolfram Mathematica (nyní ve verzi 8) umí:

- běžné numerické výpočty
- zobrazit grafy funkcí
- symbolické výpočty (úpravy algebraických výrazů, řešení rovnic, derivace, integrály,...)
- animované simulace (grafy goniometrických funkcí, kmitání oscilátoru,...)
- vyrábět kvalitní interaktivní technické dokumenty
- definovat uživatelské funkce
- a mnoho dalšího...

3.1 Jak Mathematica pracuje

Základem je tzv. *notebook* – je to soubor, do kterého se píše příkazy, poznámky, a do kterého Mathematica zapisuje výsledky.

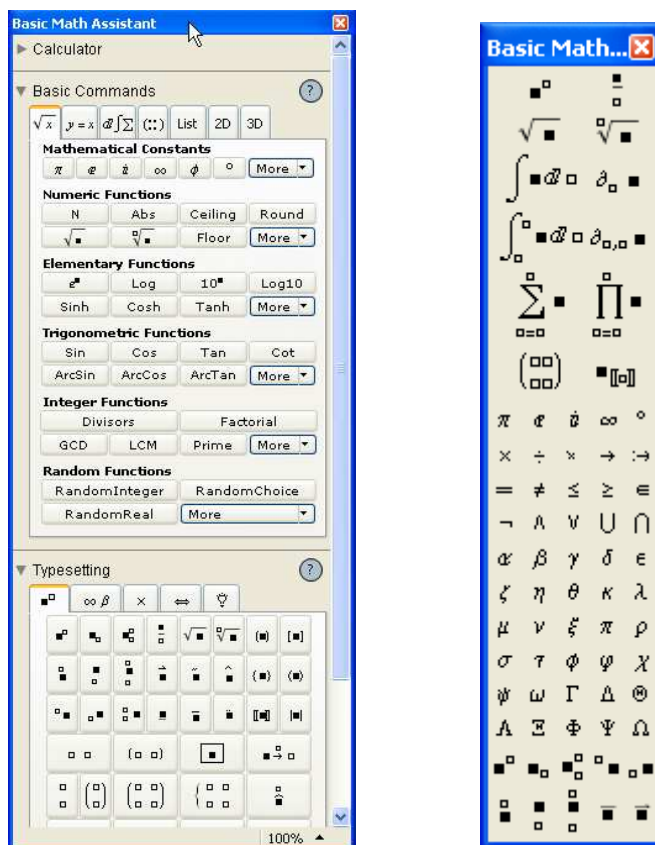
Notebook se dále dělí na *buňky*. Každá buňka může obsahovat jeden nebo více příkazů. Buňky mohou být vstupní (označené `In[cislo]`) nebo výstupní (označené `Out[cislo]`). Buňky vizuálně a funkčně oddělují text na vstupy, výstupy, grafiku, nadpisy atd.

`In[1]:= 1 + 1`

`Out[1]= 2`

Obr. 7. Buňky software Mathematica

V notebooku je uložen kompletní systém včetně sazby matematických výrazů, formátovaného textu, zvuku, grafiky, animací a hypertextových odkazů. Notebook můžeme převést do jiných formátů jako např. HTML, TeX.



Obr. 8. Palety v software Mathematica (Basic Math Input a Basic Math Assistant)

Mathematica má dvě oddělené vrstvy. S uživatelem komunikuje grafické rozhraní (tzv. GUI), které má formu různých menu a palet pro zadávání symbolů. Pro vkládání

matematický vzorců budeme nejčastěji používat paletu *Basic Math Assistant*, která umožňuje zápis zlomků, odmocnin, matic, atd. ve stejném tvaru, jak jsme zvyklí je psát ručně. Palety si můžeme vytvořit i vlastní, např. paletu nejpoužívanějších funkcí nebo si vytvořit vlastní navigační tlačítka.



Obr. 9 Hlavní menu software Mathematica

Jádro pak zpracovává zadání a interpretuje výsledky zpět do grafického rozhraní. V jádru probíhá výpočet symbolicky s maximální možnou přesností a stejně tak je i standardně interpretován výsledek. Samozřejmě je možné výsledek vypsat numericky za cenu ztráty přesnosti.

3.2 Příkazy

Mathematica obsahuje více než 1000 příkazů, další (uživatelské) lze vytvořit. Všechny systémové příkazy začínají velkým písmenem. Příkazy si není nutné pamatovat všechny. Po zadání několika počátečních písmen (funguje i po jednom písmenu) a stisknutí Ctrl+K vypíše seznam příkazů, které začínají zadanými písmeny.

3.2.1 Arimetrické operace

Základní operace jako sčítání, odčítání, násobení, dělení a umocnění se zadávají pomocí klasických znamének $+$, $-$, $*$, $/$, $^$, které se zadávají pomocí klávesnice. Operace se provádí v pořadí podle základních matematických pravidel. Pomocí závorek $($ a $)$ lze kontrolovat seskupování operací.

3.2.2 Základní funkce

- $\text{Sqrt}[x]$ - odmocnina (standardně výsledek zůstává v exaktním tvaru, pro numerický aproximovaný tvar použijeme přepínač $/N$)
- $\text{Exp}[x]$ – exponenciální funce
- $\text{Log}[x]$ – přirozený logaritmus ($\ln x$)
- $\text{Log}[z, x]$ – logaritmus o základu z
- $n!$ - faktoriál

- $\text{Sin}[x]$, $\text{Cos}[x]$, $\text{Tan}[x]$ – trigonometrické funkce
- $\text{ArcSin}[x]$, $\text{ArcCos}[x]$, $\text{ArcTan}[x]$ – inverzní trigonometrické funkce

3.2.3 Proměnné

Mathematica umí pracovat s deklarovanou proměnnou. Syntaxe deklarace je **x =hodnota**. Tato deklarace je platná pro všechny další použití proměnné x , do přepsání novou deklarací nebo vymazání deklarace přiřazením symbolu **tečka** ($x=.$) nebo použitím funkce `Clear[x]`. Proměnná, která nemá přiřazenu žádnou hodnotu, vystupuje jako symbol.

3.2.4 Funkce

Mathematica umožňuje uživateli definovat nové funkce. Používá symbol přiřazení (**$:=$**).

```
In[1]:= f[x_] := x^2
In[2]:= f[4]
Out[2]= 16
```

Obr. 10. Definice uživatelské funkce

Pro smazání definice funkce se používá funkce `Clear[f]`.

3.2.5 Komplexní čísla

Mathematica umí pracovat i s komplexními čísly. Komplexní číslo se dá napsat pomocí nástrojové palety nebo posloupnosti kláves `Esc + ii + Esc` nebo velké písmeno `I`. Obyčejné malé `i` se považuje za proměnnou.

S komplexními čísly se pracuje stejně jako s reálnými čísly, můžeme je sčítat, odčítat, násobit apod. Navíc je pro komplexní čísla definováno několik funkcí navíc:

- $\text{Re}[z]$ – vypíše reálnou část komplexního čísla z
- $\text{Im}[z]$ – vypíše imaginární část komplexního čísla z
- $\text{Conjugate}[z]$ – vypíše komplexně sdružené číslo
- $\text{Abs}[z]$ – absolutní hodnota komplexního čísla z
- $\text{Arg}[z]$ – argument komplexního čísla z

3.2.6 Vektory, matice

Vektory jsou reprezentovány seznamem prvků, např. $\{a, b, c\}$, matice seznamem řádků matice, např. $\{\{a, b\}, \{c, d\}\}$. Vektory a matice můžeme vytvořit pomocí nástrojových palet nebo nabídky **Input** → **Create Table/Matrix/Palette...** nebo složených závorek.

Pro lepší představu lze zobrazit vektory a matice v tzv. maticovém zápise použitím příkazu `MatrixForm[m]`.

Pomocí hranatých závorek můžeme přistupovat k jednotlivým prvkům vektorů (`v[[i]]` – *i*-tý prvek vektoru *v*) a matic (`m[[i, j]]` – prvek na pozici *i, j* matice *m*).

3.2.7 Funkce Table

Pro definici konkrétního vektoru, resp. matice se využívá funkce `Table`. Má široké možnosti využití a může zjednodušit práci. Příklady syntaxe:

- `Table[výraz, {n}]` – vytvoří vektor, který obsahuje *n* kopií výrazu
- `Table[fce, {i, max}]` – vytvoří vektor, který obsahuje výsledky *fce* pro *i* od 1 do *max*
- `Table[fce, {i, max, di}]` – totéž s krokem *di*
- `Table[fce, {i, m}, {j, n}]` – vytvoří matici *m*×*n*, s funkcí pro *i* od 1 do *m* a *j* od 1 do *n*

3.2.8 Tabulky

Vektory a zejména matice je někdy výhodné zobrazit ve formě tabulky. K tomu složí příkaz `TableForm`, který pracuje podobně jako `MatrixForm`. Tabulky se dají formátovat pomocí parametrů:

- **TableAlignments** – zarovnání buněk tabulky
- **TableDepth** – nastavení maximální dimenze tabulky
- **TableDirections** – vypisování dimenzí po sloupcích nebo po řádcích
- **TableHeadings** – nastavení hlavičky tabulky
- **TableSpacing** – nastavení velikosti mezery mezi řádky, resp. sloupci

3.2.9 Grafy funkcí

3.2.9.1 Funkce Plot

Funkce Plot se používá pro vykreslení dvojrozměrného grafu. V argumentu musí být zadána vykreslovaná funkce, proměnná, pro kterou je funkce vykreslována, a její meze (`Plot[fce, {x, xmin, xmax}]`). Funkce Plot může vykreslit průběh i několika funkcí do jednoho grafu. V tom případě musí být tyto funkce uzavřeny do složených závorek (`Plot[{fcel, ..., fcen}, {x, xmin, xmax}]`)

Funkce Plot má mnoho parametrů, pomocí kterých lze změnit vzhled grafu přesně podle požadavků uživatele. Všechny parametry vypíšeme pomocí funkce **Options**.

```
In[1]:= Options[Plot]
Out[1]= {AlignmentPoint → Center, AspectRatio →  $\frac{1}{\text{GoldenRatio}}$ , Axes → True,
  AxesLabel → None, AxesOrigin → Automatic, AxesStyle → {}, Background → None,
  BaselinePosition → Automatic, BaseStyle → {}, ClippingStyle → None,
  ColorFunction → Automatic, ColorFunctionScaling → True, ColorOutput → Automatic,
  ContentSelectable → Automatic, CoordinatesToolOptions → Automatic,
  DisplayFunction → $DisplayFunction, Epilog → {}, Evaluated → Automatic,
  EvaluationMonitor → None, Exclusions → Automatic, ExclusionsStyle → None,
  Filling → None, FillingStyle → Automatic, FormatType → TraditionalForm,
  Frame → False, FrameLabel → None, FrameStyle → {}, FrameTicks → Automatic,
  FrameTicksStyle → {}, GridLines → None, GridLinesStyle → {}, ImageMargins → 0.,
  ImagePadding → All, ImageSize → Automatic, ImageSizeRaw → Automatic,
  LabelStyle → {}, MaxRecursion → Automatic, Mesh → None, MeshFunctions → {#1 &},
  MeshShading → None, MeshStyle → Automatic, Method → Automatic,
  PerformanceGoal → $PerformanceGoal, PlotLabel → None, PlotPoints → Automatic,
  PlotRange → {Full, Automatic}, PlotRangeClipping → True, PlotRangePadding → Automatic,
  PlotRegion → Automatic, PlotStyle → Automatic, PreserveImageOptions → Automatic,
  Prolog → {}, RegionFunction → (True &), RotateLabel → True,
  Ticks → Automatic, TicksStyle → {}, WorkingPrecision → MachinePrecision}
```

Obr. 11. Výpis všech parametrů funkce Plot

3.2.9.2 Funkce Plot3D

Funkce Plot3D vykreslí graf funkcí dvou nezávislých proměnných. Tato funkce má opět mnoho parametrů a široké možnosti vykreslení grafu. Základní syntaxe je: `Plot3D[fce, {x, xmin, xmax}, {y, ymin, ymax}]`. Výpis všech parametrů můžeme vypsat opět funkcí **Options**.

3.2.9.3 *Funkce Show*

Jak jsme si řekli dříve, Mathematica může do proměnných ukládat téměř všechno, proto tam můžeme uložit i graf. Graf uložíme do proměnné příkazem `graf=Plot[sin[4 x], {x, 0, π}`. Uložený graf zobrazíme příkazem `Show[graf]`.

3.2.9.4 *Ostatní typy grafů*

Mathematica umí vykreslit desítky jiných grafů, např. bodový, obrysový, graf hustoty, s logaritmickým měřítkem, sloupcový graf, koláčový graf,...

3.2.10 **Derivace**

Mathematica najde derivaci téměř jakékoliv funkce. Derivaci lze řešit symbolicky (nečíselně), ale je možné najít derivaci v konkrétním bodě. Slouží k tomu funkce **D**. Její základní syntaxe je: `D[fce, x]`. Najde parciální derivaci podle **x**. Funkce `D[fce, {x, n}]` najde n-tou derivaci fce podle **x**. [11]

3.2.11 **Integrál**

Pro výpočet integrálů slouží funkce `Integrate`. Pokud nezadáme meze, vypočítá se neurčitý integrál, v opačném případě se vypočítá integrál určitý. Syntaxe:

- Pro neurčitý integrál `Integrate[fce, x]`
- Pro určitý integrál `Integrate[fce, {x, xmin, xmax}]`

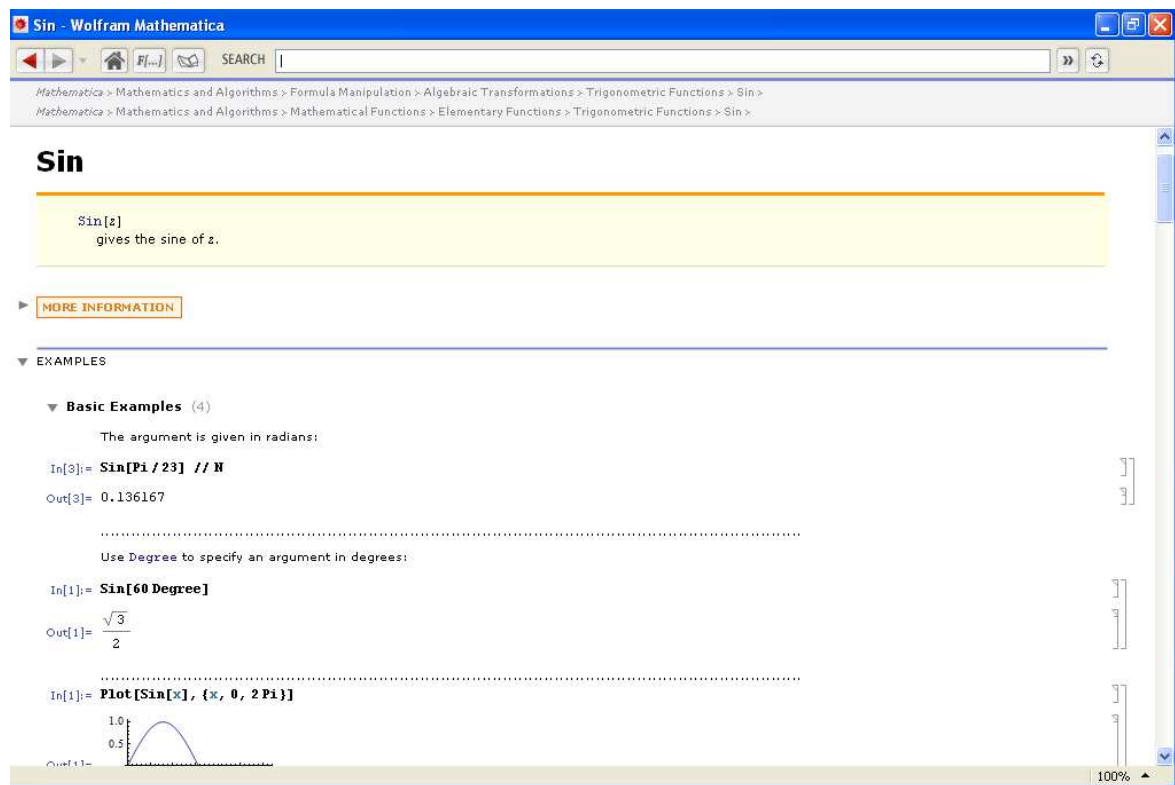
3.3 **Text**

V notebooku můžeme psát i dlouhé texty (články, skripta, apod.), lze je formátovat podobně jako v textových editorech, a přitom využít matematických a programovacích schopností softwaru Mathematica – můžeme vytvořit interaktivní notebook. Notebook se dá potom vyexportovat do různých formátů, např. `*.html`.

3.4 **Nápověda**

Nápověda v software Mathematica je fantasticky zpracovaná. Obsahuje kompletní dokumentaci pro všechny funkce, jejich syntaxi (u řady příkazů je možných více zápisů), popis a hlavně příklady. Každý příklad si může uživatel upravit, a zjistit „co to udělá“, aniž

by musel mít strach, že nápovědu poškodí. Nápověda obsahuje také úplný text The Mathematica Book jako plně indexovaný notebook.



Obr. 12. Interaktivní nápověda v software Mathematica

4 TVORBA DOKUMENTŮ VE FORMÁTU HTML

HTML (HyperText Markup Language) je programovací jazyk internetových stránek. Navrhl ho v roce 1990 Tim Berns-Lee při práci na informační systému pro CERN, zároveň napsal první webový prohlížeč, který nazval *WorldWideWeb*. V roce 1991 CERN zprovoznil svůj první web.

V říjnu 1994 založil Tim Berns-Lee při MIT (Massachusetts Institute of Technology) World Wide Web Consortium (W3C), které má pomáhat rozvíjet protokoly a zajistit dlouhodobý růst webu.

HTML je aktuálně ve verzi 5, která má oproti minulé verzi velkou podporu multimédií.

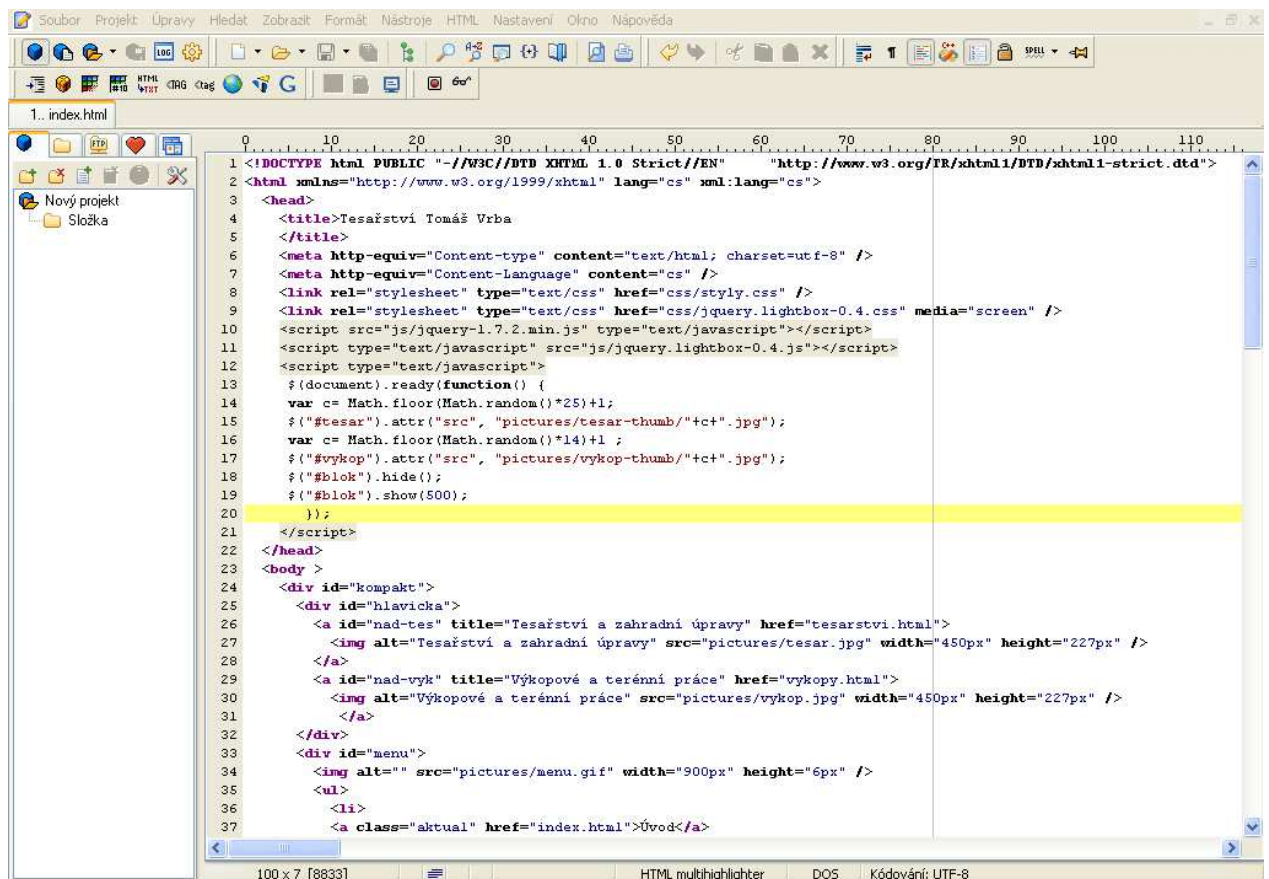
4.1 Vytvoření zdrojového kódu

K vytváření dokumentů ve formátu HTML můžeme použít tzv. *WYSIWYG* (What you see is what you get – co vidíš, to dostaneš) *editory*. Webovou stránku tvoříme podobně jako dokument ve Wordu, tzn. píšeme text, formátujeme ho pomocí menu, vkládáme obrázky, atd. Program automaticky vytváří zdrojový kód. Problémem je sporná kvalita výsledného zdrojového kódu a jeho problematická kontrola.

Druhou možností je použití jakéhokoliv textového editoru, do kterého budeme zapisovat kromě textu k zobrazení i **tagy** (česky se jim říká také značky, jsou to vlastně příkazy jazyka HTML), uložíme jako čistý text a změníme příponu souboru na **.html*.

Třetí možností je použít speciální editory HTML dokumentů, které fungují jako textové editory, ale jsou rozšířeny pro práci s tagy a zvýrazňují syntaxi jazyka HTML, umí zobrazit náhled webové stránky, apod.

Zdrojový kód se dělí na dvě hlavní části: hlavičku a tělo. Hlavička i tělo musí být uzavřeny mezi tagy `<html>` a `</html>`. Mimo tyto tagy, jako úplně první řádek se zapisuje tag `<!DOCTYPE>`. Uvádí specifikaci DTD, podle standardů je povinná, ale prohlížeče stránku zobrazí i bez tohoto tagu.



Obr. 13. Okno editoru se zvýrazňováním HTML kódu

4.1.1 Hlavička

Hlavička je část zdrojového textu mezi tagy `<head>` a `</head>`. Tento tag může obsahovat několik nepovinných tagů, a to:

- `<title>` Nadpis `</title>` - nastavuje „Nadpis“ jako titulek stránky
- `<base>` - nastavuje základní cestu pro relativní odkazy ve stránce
- `<link>` - slouží ke spojení s jiným souborem, např. CSS stylem. Má několik atributů (rel, href, type,...)
- `<meta>` - obsahuje informace o stránce, např. kódování znaků
- `<script>` - slouží k připojení externího skriptu (např. JavaScript)

Příklad kompletní hlavičky:

```

<head>
  <meta http-equiv="content-type"
content="text/html; charset=windows-1250">
  <title>Titulek stránky</title>

```

```
<link rel="stylesheet" href="style/muj-styl.css"
type="text/css">
</head>
```

4.1.2 Tělo (body) dokumentu

Tělo dokumentu je část zdrojového textu mezi tagy `<body>` `</body>`. Mezi tyto tagy zapisujeme kód stránky, obsahující vlastní text k zobrazení a formátovací příkazy – tagy. Také tag `<body>` může obsahovat několik nepovinných tagů (např. `bgcolor`, `background`, `link`, `vlink`,...), ale dnes se nahrazují použitím CSS vlastností.

4.2 Řádkové značky HTML

Formátování webových stránek ve formátu HTML můžeme rozdělit na dvě skupiny:

- Fyzické formátování – používá se pro nastavení vzhledu, tj. velikosti, barvy, dekorace, atd.
- Logické formátování – používá se pro logické dělení stránky podle obsahu [5]

4.2.1 Fyzické formátování HTML

Formátovat dokumenty HTML víceméně dvěma způsoby:

- pomocí tagů – např. text v kurzívě se vypsal pomocí tagů `<i>` a `</i>`
- pomocí CSS – používá se tag `<style>` a atribut `style`.

Formátování pomocí tagů je již zastaralé a jejich pomocí nejde udělat všechno, co lze pomocí CSS, proto se budu dále věnovat pouze formátování pomocí CSS. CSS je zkratka Cascading Style Sheets (česky „kaskádové styly“).

CSS poprvé implementoval Microsoft do Internet Exploreru 3.0 v roce 1996. Pomocí CSS můžeme definovat kromě barvy, písma a velikosti i další věci, např. rámeček, odrážky, okraje,...

Pomocí CSS můžeme velmi zjednodušit a urychlit práci:

- jsou k dispozici nové formátovací prvky (rámečky, okraje, barvy,...), které se pomocí samotného HTML tvořily jenom těžce a neobratně
- pomocí CSS se jednoduše deklaruje styl pro celý web, a jednoduše se mění

- šetří místo a práci, jeden soubor se použije pro celý web
- může se nastavit chování i standardních tagů
- pomocí JavaScriptu lze dynamicky měnit styly pro jednotlivé komponenty (CSS a JavaScript se spojují identifikátorem ID)

Pro fyzické formátování se používají následující značky: b (tučný text), basefont (nastavení základní písmo), big (zvětšení písma o jeden stupeň), blink (blikající písmo), font (nastavení písma), i (kurzíva), nobr (zakazuje zalomení řádku), s (přeškrtnuté písmo), small (zmenšení písma o jeden stupeň), strike (přeškrtnuté písmo), sub (dolní index), sup (horní index) a u (podtržení). Všechny tyto značky jsou nahraditelné kaskádovými styly a je doporučeno je tak nahradit.

4.2.2 Logické formátování HTML

Tagy logického formátování formátují text i fyzicky, ale i logicky rozdělují text. Jedná se o tyto značky: abbr (zkratka, její vysvětlení se zapisuje do parametru title), acronym (zkratka, která se po písmenech, ale jako slovo), cite (řádková citace, většinou se vykresluje kurzívou), code (úsek kódu, vypisuje se neproporcionálním písmem), del (odstraněný text, vykresluje se přeškrtnutým fontem), ins (přidaný text, vykreslí se jako podtrhnutý), kdb (zápis z klávesnice, neproporcionální font), samp (výstup z programu, taktéž neproporcionální font), span (vytváří blok textu, kterému lze přidělit styl pomocí atributů style, id nebo class; sám nijak neformátuje text), strong (zvýraznění, tučné písmo), q (citace, atribut cite obsahuje zdroj citace) a var (proměnná, kurzíva).

Tyto značky nemají alternativu v CSS a nelze je tedy tímto způsobem nahradit.

4.3 Blokované značky HTML

Blokované značky se od řádkových liší zejména tím, že jejich ukončovací tag zalomí text na další řádek. Mezi blokované řadíme tyto značky: p (odstavec, blok textu, který má mezeru zdola i shora, vlastnosti lze nastavovat pomocí CSS), h1, h2, h3, h4, h5, h6 (nadpisy, standardně je h1 největší a postupně se zmenšují, vlastnosti lze změnit pomocí CSS), br (přechod na nový řádek), div (blokový span, sám o sobě nijak neformátuje stránku; je důležitý pro formátování pomocí CSS, pomocí divů se tvoří například layout stránky), hr (vodorovná čára), pre (zobrazuje všechny zapsané znaky i všechny mezery, řádkování,

tabulátory, ušetří spoustu práce při psaní zdrojového kódu se stromovou strukturou) a blockquote (víceřádková citace).

4.4 Seznamy

Seznam je několik položek, které se zapisují do řádků pod sebou. Seznam v HTML může být číslovaný nebo s odrážkami.

4.4.1 Číslovaný seznam

Jednotlivé položky seznamu se zapisují mezi tagy `` a ``. Každá položka musí být uzavřena do tagů `` a ``. Je možné nastavit počáteční číslo seznamu (atribut `start`) a typ číslování. Typ číslování může být „**A**“ – velká abeceda, „**a**“ – malá abeceda, „**i**“ – malé římské číslice, „**I**“ – velké římské číslice, „**1**“ – arabské číslice a zapisuje se do atributu **type**.

4.4.2 Odrážkový seznam

Jednotlivé položky seznamu se zapisují mezi tagy `` a ``. Každá položka musí být uzavřena stejně jako u číslovaného seznamu do tagů `` a ``. Je možné nastavit typ odrážky v atributu `type`. Možné hodnoty jsou „**square**“, „**circle**“ a „**disc**“, pro odrážky ve tvaru čtverečku, kolečka a puntíku. Pomocí CSS lze seznam ještě vylepšit – je možné nastavit obrázek místo odrážek.

4.4.3 Vnořené seznamy

Seznamy lze libovolně vnořovat, a to tak, že mezi tagy `` `` nebo `` `` se vloží další tagy `` `` nebo `` ``.

4.4.4 Další typy seznamů

Seznam definic – mezi tagy `<dl>` `</dl>` jsou vždy dvojice tagů `<dt>` `</dt>` a `<dd>` `</dd>`. Mezi tagy `<dt>` a `</dt>` je **pojmem**, který chceme definovat a mezi tagy `<dd>` a `</dd>` je **definice pojmu**.

Tagy `<dir>` `</dir>` - tento seznam se chová jako jako `` ``; není třeba ho používat

Tagy `<menu>` `</menu>` - tento seznam se chová jako jako `` ``; není třeba ho používat

4.5 Odkazy

Odkaz se vytváří pomocí značky `<a>` ``. Text odkazu se nachází mezi těmito tagy. Může to být i obrázek. Cíl odkazu se zapisuje do atributu **href**. Může to být absolutní adresa (www.seznam.cz), relativní adresa nebo odkaz na element ve stránce.

4.6 Tabulky

Tabulka se zapisuje mezi tagy `<table>` a `</table>`. Vlastnosti tabulky se dají zapsat pomocí atributů, nebo CSS (což je více doporučováno). Tabulka se skládá z jednotlivých řádků, které zapisujeme mezi tagy `<tr>` a `</tr>`. Obsah jednotlivých buněk je mezi tagy `<td>` a `</td>`. Buňky je možné slučovat po řádcích pomocí atributu „**rowspan**“ a po sloupcích „**colspan**“.

V prvním řádku tabulky je možné použít místo značek `<td>` `</td>` značky `<th>` a `</th>`. Text mezi těmito značkami se považuje za hlavičku tabulky a bude naformátován jiným stylem.

Mezi tagy `<caption>` `</caption>` se zapisuje nadpis tabulky. Tagem `<col />` se formátuje sloupec tabulky.

II. PRAKTICKÁ ČÁST

5 DOKUMENTY VE FORMÁTU HTML OBSAHUJÍCÍ MATEMATICKÉ VZORCE

Protože se matematické vzorce podobají více obrázkům než textu, proto s nimi má formát HTML pracující s textem velké problémy. Protože formát HTML vznikl na akademické půdě, kde se se vzorci často pracovalo, začaly se hledat možnosti, jak vzorce do HTML zakomponovat.

Nejprve se tento problém řešil vkládáním bitmapových obrázků se vzorci. Toto na první pohled elegantní řešení má několik zásadních nedostatků:

- při větším množství vzorců roste množství přenášených dat, zpomaluje se načítání stránky
- nutnost použít speciální program a grafický editor, nekomfortní editace
- nemožnost automatické interpretace dat
- nemožnost dynamické tvorby obsahu stránek

Z těchto důvodů (zejména toho prvního – v prehistorické době internetu se počítal každý přenesený bit) začaly vznikat javascriptové knihovny, které umožňují grafické zobrazení zapsaného textu. Jedna z prvních byla knihovna **WebEQ**, která se stala komerční. Nyní už není dále vyvíjena, jejím nástupce se jmenuje **MathFlow**. Nejznámější knihovnou zdarma je **jsMath**. Tato knihovna nejen používá stejnou syntaxi, ale i stejné fonty jako TeX, tzn. i výstup je téměř identický jako z TeXu. **jsMath** využívá nativní fonty, takže při změně velikosti textu, při tisku v rozlišení tiskárny nemusí uživatel čekat na stažení desítek obrázků. Stejně tak autor stránek nemusí tyto obrázky vytvářet.[12]

V roce 1995 organizace W3C uznala, že je nemožnost vkládání vzorců standartním způsobem opravdu vážným problémem. Objevilo se několik návrhů jak tento problém řešit. Prvním návrhem bylo rozšíření HTML (tehdy ve verzi 3.0) o nové tagy, které umožní formátovat vzorce. Později se rozhodlo, že vznikne obecný mechanismus a vzorce budou jenom jednou s podmnožin. Nakonec vznikl XML a jeho podmnožina **MathML** (Mathematical Markup Language). Poslední revize má označení **MathML 3.0** (je z října 2010). Značky jazyka **MathML** se zapisují podobně jako značky jazyka XHTML (oba jazyky jsou podskupinou XML). Protože se v zápisu kódu často chybuje, vznikla spousta doplňků, které ulehčují tento zápis podle standardu. Velkým problémem je omezená

podpora prohlížečů, 100% podpora je pouze u prohlížeče *Amaya* (experimální prohlížeč vyvíjený organizací W3C) a o něco slabší u prohlížečů postavených na jádře *Gecko* (Mozilla Firefox). Prohlížeč Microsoft Internet Explorer tento jazyk téměř ignoruje. [13]

5.1 Vkládání vzorců jako obrázky

Vytvořil jsem první kapitolu skript pomocí této první možnosti, kterou formát HTML nabízí. Zrovna v této kapitole není mnoho vzorců, tak mne zajímalo, jak vzroste datová náročnost. **Stránka** sama o sobě má velikost **32 kB**. **Obrázky grafů**, které se budou používat při každé možnosti tvorby, mají celkovou velikost **24,1 kB**. **Obrázky vzorců**, které bylo potřeba vytvořit, mají **135kB**.

Při tvorbě stránky jsem narazil na problém, co považovat za vzorec, který musí být nahrazen obrázkem, a co ještě za text, který se dá interpretovat pomocí ASCII znaků. Pokusil jsem se použít standartní HTML a ASCII znaky pro zápis intervalů a základních funkcí, tj. $\langle 1,3 \rangle$; $\langle x_i, x_{i+n} \rangle$; $f(x)=x^2$; apod.

`<p>Dělení intervalu <a,b> je soubor vzestupně řazených hodnot
</p>`

Dělení intervalu $\langle a,b \rangle$ je soubor vzestupně řazených hodnot

Obr. 14. Vložení intervalu pomocí standartních znaků ASCII

`<p>Dělení intervalu je soubor vzestupně řazených hodnot
</p>`

Dělení intervalu $\langle a,b \rangle$ je soubor vzestupně řazených hodnot

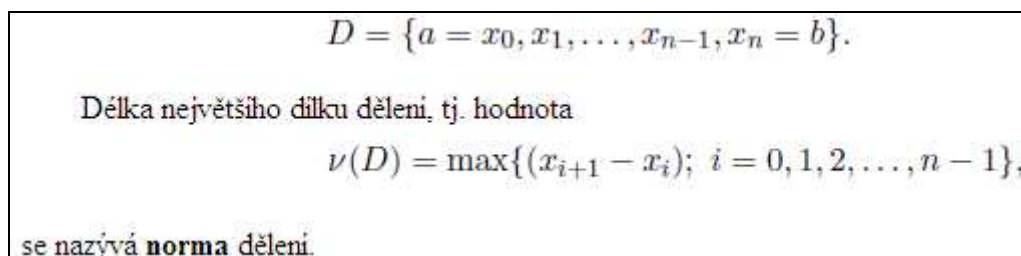
Obr. 15. Vložení intervalu pomocí obrázku

Použitím znaků ASCII je možné ušetřit poměrně velké množství přenesených dat, ale na první pohled to není ono. Pro upřesnění: na naší vzorové kapitole 8.1 ušetříme v zápisu kódu necelé 3kB (zápis pro vložení obrázku bývá většinou delší než zápis tagů pro úpravu písma). Ale musíme přenést o 55kB obrázků více.

Rovnice lze vytvořit například v programu MS Word, který obsahuje pokročilý editor rovnic, který bohužel neumožňuje uložení rovnice jako obrázku. Protože jsem chtěl, aby si byly znaky ve všech příkladech, co nejpodobnější použil jsem pro zápis rovnice TeXový editor TexnicCenter a po vyrenderování na obrazovku jsem použil program Gadwin

PrintScreen, který rozšiřuje systémový příkaz „printscreen“ o možnost výběru části obrazovky. Tento výřez jsem vložil do bitmapového editoru Gimp, ve kterém jsem vzorec dále ořezal, vymazal pozadí, a překonvertoval do datově úsporného formátu *.png, který vkládám do webové stránky pomocí tagu . Velkou nevýhodou celého postupu je, že v případě chyby ve vzorci, kterou zjistím a po vložení do stránky, je nutné celou proceduru opakovat.

```
<img id='r81' class='obrstred' src='../pictures/8-1/r81.png'
alt='' /><p>Délka největšího dílku dělení, tj. hodnota</p>
<img id='r82' class='obrstred' src='../pictures/8-1/r82.png'
alt='' /> <p class='prikklad'>se nazývá <strong>norma</strong>
dělení. </p>
```



Obr. 16. Vložení rovnic jako obrázků

Při vkládání obrázků je obtížné jejich centrování z důvodu jejich rozdílné velikosti. Jako ideální se jeví pozicování obrázků každého zvlášť pomocí CSS.

Posledním a dost zásadním problémem je pozadí obrázků, které je potřeba nastavit na průhledné, aby při použití jiného než bílého pozadí nevznikl nevzhledný bílý obdélník se vzorcem. Bohužel ne vždy se to podaří dokonale, jak můžeme vidět na obrázku 17.

$$\lim_{n(D) \rightarrow 0} S(f, D) = \int_a^b f(x) dx.$$

Obr.17. Vložený obrázek na nebílém pozadí

Z těchto důvodů není používání tohoto způsobu vkládání nejvhodnější. Hodí se pouze pro stránky s bílým pozadím, na stránky, ve kterých je pouze několik vzorců (z důvodu obtížnějšího pozicování a datové náročnosti). Výhodou je, že se nemusíme učit další syntaxi zadávání vzorců.

5.2 Vkládání vzorců pomocí Javascriptu

Pro porovnání jsem vytvořil stejnou stránku pomocí javascriptové knihovny jsMath. Tato knihovna používá k zápisu vzorců TeX formát.

5.2.1 Příprava

Pro použití na svých stránkách musíme knihovnu rozbalit na server s našimi stránkami. A do každého souboru, který používá tuto knihovnu, ji musíme „nalinkovat“ HTML tagem `<script language="JavaScript" type="text/javascript" src="cesta/jsMath-3.6e/easy/load.js"></script>`, kde *cesta* nahradíme cestou k souboru na serveru. Jedná se o startovací skript knihovny, který knihovnu nastaví a spustí.

Aby bylo možné používat speciální znaky a zobrazovat je jako písmo (ne jako obrázek), je nutné nainstalovat speciální fonty, které tyto znaky zobrazují. Pokud knihovna nenajde při zobrazování stránky tyto fonty, vypíše hlášení (jde potlačit nebo úplně změnit) a použije místo nich obrázky. Zobrazování stránky se ale zpomalí. Proto se doporučuje nabídnout návštěvníkovi stránky tyto fonty ke stažení. Samozřejmě je možné vytvořit vlastní fonty a používat je.

5.2.2 Vlastní zápis vzorců

Vlastní vzorce se zapisují do HTML elementů `<DIV></DIV>` nebo ``, kterým se přiřadí třída „**math**“. Knihovna jsMath prochází celou stránku a pokud narazí na blok této třídy, nahradí standartní font stránky speciálním fontem, popř. obrázky znaků. Celý zápis vzorce může vypadat například takto:

```
<div class="math"> D=\{a=x_0, x_1, \ldots, x_{n-1}, x_n =b\}</div>
```

```
<div class="math"> vytváří oblast třídy „math“
```

```
</div> ukončuje oblast třídy „math“
```

Všechno mezi tagy se považuje za vzorec, který se má zobrazit ve speciálním formátu.

Zápis uvnitř bloku vypadá po vykreslení takto:

$$D = \{a = x_0, x_1, \dots, x_{n-1}, x_n = b\}$$

Obr.18. Vzorec vykreslený pomocí jsMath

5.2.3 Vlastnosti stránky s použitím jsMath

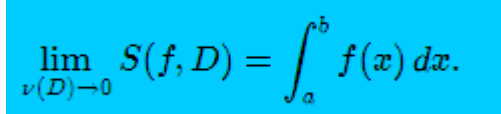
Ze všeho nejdříve mne zajímala velikost. Samotná stránka mírně narostla. Nyní má 38kB, oproti původním 32kB. Tzn. že zápisem vzorců sice vzrostla velikost samotného HTML souboru, ale nemusíme načítat další obrázky. Takže jsme ušetřili více než 130kB. Problém vznikne při načítání první stránky, která používá jsMath, protože knihovna se musí taktéž načíst do paměti. Celá knihovna má cca. 600 kB, nikdy se ale nenačítá celá. Většinou se načítá asi polovina, záleží na tom, kolik používáte na svých stránkách plug-inů a rozšíření knihovny. Načítá se vždy jenom při prvním spuštění stránky s knihovnou jsMath, pak zůstává v paměti a pouze se spouští.

300kB se může zdát hodně, ale na stránce, která nemá mnoho vzorců, jsme oproti použití bitmapových obrázků ušetřili téměř polovinu. Při zvýšení počtů vzorců a načtení většího množství stránek se to již vyplatí.

Toto všechno platí, pokud má návštěvník stránek nainstalované TeX fonty, které jsMath používá pro zobrazování vzorců. Pokud tomu tak není, zobrazuje jsMath vzorce pomocí obrázků, které jsou uloženy na serveru, a proto datová náročnost roste. Jeden znak má průměrnou velikost 150B, takže i tak bude datová náročnost daleko menší než použití bitmapových obrázků.

Elementy <DIV> třídy „math“ jsou centrovány na střed, v případě potřeby je možné je pomocí CSS pozicovat unikátními identifikátory. Nebo je můžeme všechny posunout vlevo a unikátními identifikátory vystředit pouze některé.

Protože používáme nativní fonty, popř. z nich vytvořené obrázky nemáme žádný problém při umístování vzorců na barevné pozadí (nikde nám neprosvítá bílá). Při zvětšování textu přes menu prohlížeče se zvětšují i vzorce.


$$\lim_{\nu(D) \rightarrow 0} S(f, D) = \int_a^b f(x) dx.$$

Obr.19. Vložený vzorec na nebílém pozadí

Z výše uvedených důvodů se tento způsob vkládání hodí na stránky, které obsahují mnoho vzorců (aby se smazala nevýhoda stahovat knihovnu), a na stránky, u jejichž návštěvníků se dá předpokládat nainstalované TeX fonty (pro rychlejší zobrazování – nemusí se stahovat obrázky znaků – zvýšení datové náročnosti).

5.3 Vkládání vzorců pomocí jazyka MathML

Poslední možností jak tvořit stránky s matematickými vzorci psaním kódu je využití jazyka MathML. MathML je stejně jako HTML značkovací jazyk, jehož specifikaci vyvíjí a udržuje společnost W3C. I když je již ve verzi 3.0, je jeho podpora u prohlížečů velmi slabá.

Podobně jako u funkce f lze odhadnout obsah obrazce P , můžeme pak psát

$$\sum_{i=0}^{n-1} m_i \cdot |x_{i+1} - x_i| \leq P \leq \sum_{i=0}^{n-1} M_i \cdot |x_{i+1} - x_i|$$

Dolní, resp. horní odhad obsahu obrazce nazýváme **dolním**, resp. **horním součtem** pro funkci $f(x)$ a dělení D , a značíme

$$L(f, D) = \sum_{i=0}^{n-1} m_i \cdot |x_{i+1} - x_i|, \quad U(f, D) = \sum_{i=0}^{n-1} M_i \cdot |x_{i+1} - x_i|$$

Obr. 20 Internet Explorer tagy jazyk MathML úplně ignoruje

hledaný obsah obrazce P , můžeme pak psát

$$\sum_{i=0}^{n-1} m_i \cdot |x_{i+1} - x_i| \leq P \leq \sum_{i=0}^{n-1} M_i \cdot |x_{i+1} - x_i|$$

Dolní, resp. horní odhad obsahu obrazce nazýváme **dolním**, resp. **horním součtem** pro funkci $f(x)$ a dělení D , a značíme

$$L(f, D) = \sum_{i=0}^{n-1} m_i \cdot |x_{i+1} - x_i|, \quad U(f, D) = \sum_{i=0}^{n-1} M_i \cdot |x_{i+1} - x_i|$$

Obr. 21. Chrome (jádro Webkit) se snaží značky interpretovat

je „horním odhadem“ obsahu obrazce. Označíme-li hledaný obsah obrazce P , můžeme pak psát

$$\sum_{i=0}^{n-1} m_i \cdot |x_{i+1} - x_i| \leq P \leq \sum_{i=0}^{n-1} M_i \cdot |x_{i+1} - x_i|$$

Dolní, resp. horní odhad obsahu obrazce nazýváme **dolním**, resp. **horním součtem** pro funkci $f(x)$ a dělení D , a značíme

$$L(f, D) = \sum_{i=0}^{n-1} m_i \cdot |x_{i+1} - x_i|, \quad U(f, D) = \sum_{i=0}^{n-1} M_i \cdot |x_{i+1} - x_i|$$

Obr.22. Zobrazení v experimentálním prohlížeči Amaya

můžeme pak psát

$$\sum_{i=0}^{n-1} m_i \cdot (x_{i+1} - x_i) \leq P \leq \sum_{i=0}^{n-1} M_i \cdot (x_{i+1} - x_i)$$

Dolní, resp. horní odhad obsahu obrazce nazýváme **dolním**, resp. **horním součtem** pro funkci $f(x)$ a dělení D , a značíme

$$L(f, D) = \sum_{i=0}^{n-1} m_i \cdot (x_{i+1} - x_i), \quad U(f, D) = \sum_{i=0}^{n-1} M_i \cdot (x_{i+1} - x_i)$$

Obr. 23. Zobrazení v prohlížeči Mozilla (jádro Gecko)

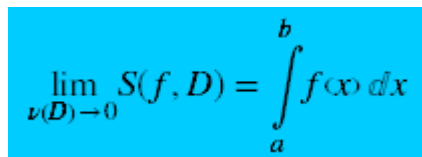
Protože MathML vychází ze striktního XML, jsou všechny tagy párové. Vzorce se vkládají do mezi tagy MathML $\langle \text{math} \rangle$ a $\langle / \text{math} \rangle$. Vše co je mezi těmito tagy chápe prohlížeč

jako vzorec. Dále se vkládají tagy <math><math> - pro řádek, <math><math> - pro proměnnou, <math><math> - pro čísla, <math><math> - pro operátory a další pro zlomky, mocniny,...

Těchto tagů je poměrně mnoho (téměř každý znak se píše mezi dva tagy), proto nám narostla velikost. Stránka má nyní 159kB, ale nepotřebujeme nahrávat nic dalšího. Je to podstatně více než při použití javascriptu, ale šetříme data, když nemusíme přenášet samotné soubory javascriptu. Je to i více než při použití obrázkových bitmap.

Opět můžeme pozicovat každý element pomocí CSS.

Při použití nativních fontů nemáme problém se zobrazením na barevném pozadí.


$$\lim_{\nu(D) \rightarrow 0} S(f, D) = \int_a^b f(x) dx$$

Obr. 24. Vložení vzorce na barevné pozadí

6 EXPORT Z LATEXU

K exportu z LaTeXu je možno použít několik různých programů:

- LaTeX2HTML
- TeX4ht
- LaTeXML
- a další

Prvním a nejspíš nejzásadnějším problémem, který odradí většinu uživatelů je nutnost instalace těchto programů s překladem ze zdrojových kódů, které často končí chybovou hláškou. Nejrychlejší možností je stažení kompletní distribuce, která již některý program obsahuje. I když se konverze spustí, neznamená to, že máme vyhráno.

6.1 LaTeX2HTML

Tento převodník napsal Nikos Drakos, později ho doplnila spousta přispěvatelů. Je to obrovský skript v Perlu, který pomocí dvips, Ghostscriptu a knihovny netpbm konvertuje dokumenty LaTeXu do HTML. Tento převodník již není dále vyvíjen. Některé odkazy z domovské stránky již nefungují.

Při zpracování vytvoří LaTeX2HTML nový adresář se jménem souboru, a zapisuje do něho výsledný kód HTML a generované obrázky ve formátu *.gif*.

Program pracuje na stejném principu jako samotný LaTeX jen místo souboru *.dvi* zapisuje do souborů *.html* (samozřejmě jinou syntaxí). Oba jazyky jsou si do jisté míry dost podobné, i když podobnost brzy dosahuje limitů. Mnoho vlastností LaTeXu není v HTML dostupných.[14]

6.2 TeX4ht

Tento program napsal Eitan M. Gurari. Vytváří téměř stejný výstup jako LaTeX2HTML, ale jde na to jiným způsobem. Nezpracovává přímo zdrojový text, ale již zpracovaný soubor *.dvi*. Standardní soubor *.dvi*, ale neobsahuje žádné informace o značkování, proto je nutné vytvořit speciální soubor *.dvi* odpovídajícím balíčkem TeX4ht.sty. [6]

Konverze tímto programem byla jediná, která se mi povedla spustit, bylo to zejména tím, že tento program je implementován v distribuci MiKTeX, kterou používám. Bohužel si program neporozuměl s obrázky, které se ve zdrojovém textu vykreslovaly pomocí nadstavby TikZ a tyto obrázky nevyexportoval. Dalším problémem je, že místo všech písmen „á“ se vykreslila pouze „ ´ “ (čárka).

Na přiloženém CD jsem dokument nechal přesně tak, jak byl vyexportovaný. Samozřejmě se dá ručně opravit, přidat odkazy na obrázky, upravit formátování apod.

8.1 Integrovní - „sčítání“ mnoha malých příspěvků

Studijní cíle

1. Pochopit proces integrování tzv. proužkovou metodou.
2. Umět vysvětlit odvození Riemannova integrálu - pomocí metody horních a dolních součtů a pomocí metody integrálních součtů.

Motivace

Abychom pochopili podstatu integrování, vyjděme z formulace následujícího geometrického problému: Necht' je dána spojit' nezáporná funkce $f(x)$ v intervalu (a, b) .

Graf této funkce spolu s přímkami o rovnicích $x = a$, $x = b$ a osou x vymezí v kartézské soustavě souřadnic rovinný útvar. Naším úkolem je zjistit obsah tohoto rovinného obrazce.

$xy = f(x), x = a, x = b$

Obr. 8.1.1:

Z pohledu na Obr. 8.1.1 jistě plyne, že stanovit číslo vyjadřující obsah rovinného obrazce je naprosto korektní úvaha. Z minulých školních let určitě ještě znáte nějaké vzorce pro výpočet obsahů (kruh, obdélník, lichoběžník, trojúhelník), které bychom snad mohli nějakým způsobem využít. Pokusme se nejdříve stanovit obsah obrazce alespoň přibližně. Kdybychom si nakreslili obrazec na milimetrový papír a spočítali všechny čtverečky o obsahu 1mm^2 , které se do našeho obrazce vejdou, měli bychom zhruba obsah obrazce vypočítané. Uvědomíme-li si, že náš útvar není zcela obecný, ale je omezen hned třemi přímkovými útvary a teprve čtvrtý útvar je obecnou křivkou, vidíme, že takové „měření“ lze velice snadno zjednodušit. Nakreslíme útvar na milimetrovou síť tak, aby osa x splývala se stranou některé řady čtverečků. Pak stačí jen sečíst obsahy sloupečků, obdélníčků (proužků) s milimetrovou podstavou, které jsou v obrazci obsaženy. Proto tuto metodu nazveme „proužkovou metodou“. Na Obr. 8.1.2 je proužková metoda naznačena.

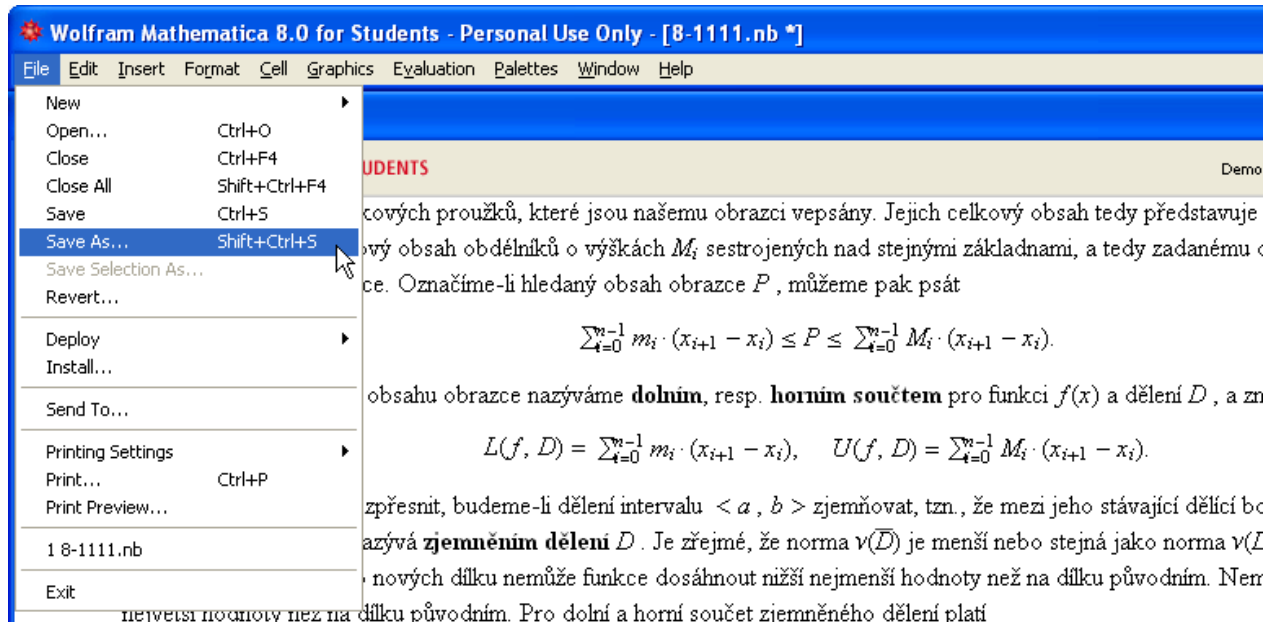
Obr. 25. Export z LaTeXu pomocí TeX4ht

6.3 LaTeXML

Tento program napsal Bruce R. Miller. Je napsán opět v Perlu. Exportuje TeXový soubor do souborů rodiny XML. Jak bylo řečeno výše, do této skupiny patří i jazyky MathML a XHTML. Na webových stránkách <http://dlmf.nist.gov/LaTeXML> je dost podrobně popsána instalace, která se nepodařila, i když jsem postupoval krok za krokem podle návodu. [15]

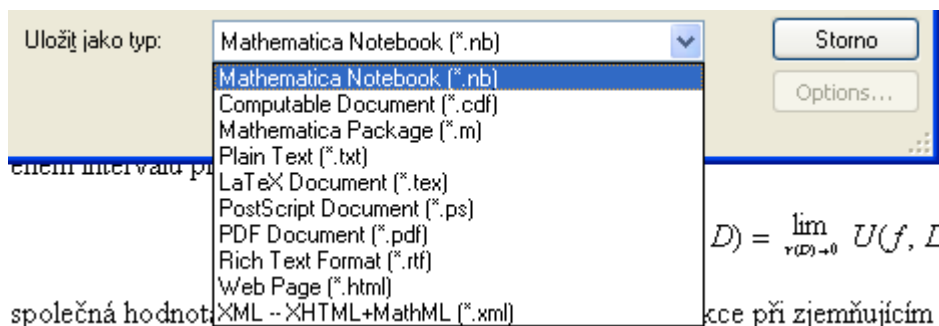
7 EXPORT ZE SOFTWARE MATHEMATICA

Po vytvoření notebooku dle naší představy v software Mathematica můžeme v menu „File“ vybrat volbu „Save As...“.



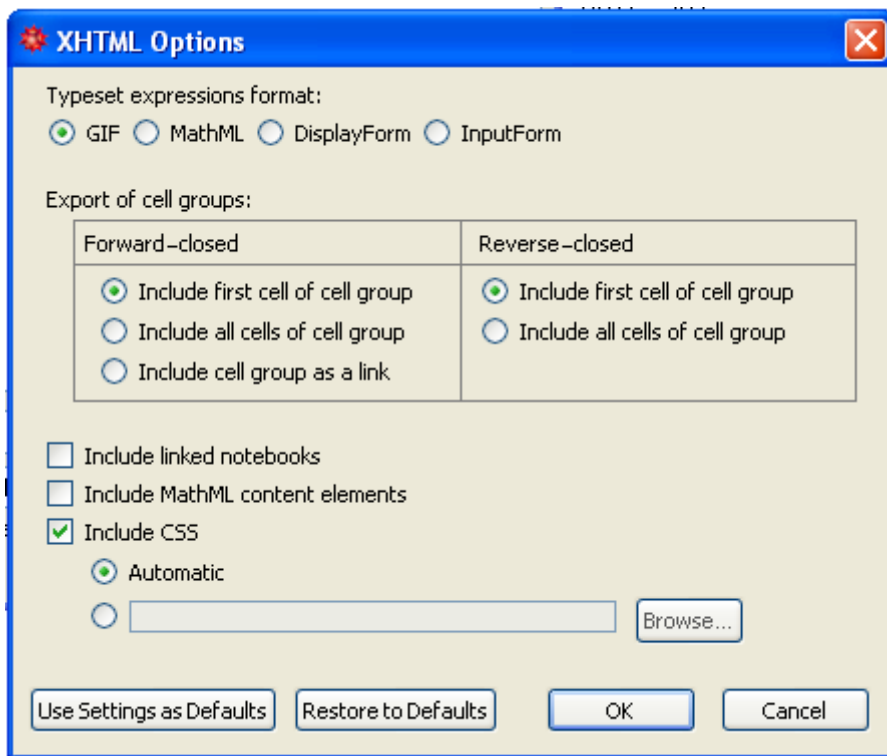
Obr. 26. Wolfram's Mathematica – Save As...

Otevře se nám systémová nabídka, ve které můžeme zvolit místo uložení souboru, jeho jméno a jeho typ.



Obr. 27. Volba formátu exportu

Nás nyní zajímá možnost uložení jako webová stránka – *Web Page (*.html)*. Po výběru se nám umožní stisknutí tlačítka *Možnosti (Options...)*, kde jsou skryty možnosti exportu.



Obr. 28. Možnosti exportu do html Wolfram's Mathematica

Jako první je důležitá volba, jak vysázet vzorce. Jako první je možnost použití bitmapových obrázků ve formátu **.gif*. Tento formát obrázků má podobné vlastnosti jako **.png*. Druhou možností je použití jazyka MathML. Pro poslední dvě volby jsem nenašel kompatibilní prohlížeč, který by zobrazil vyexportované vzorce v pořádku.

Poslední skupinu voleb používáme pro připojení dalších souborů. První volba připojí odkazované notebooky (soubory software Mathematica), pokud používáme v notebooku odkazy na jiný notebook. Druhá volba připojí obsahové prvky MathML. Poslední volbou připojíme soubor s kaskádovými styly CSS. Volba Automatic vytvoří nový soubor v adresáři HTMLFiles a připojí ho. Další možností je zvolit si vlastní umístění.

Vyexportovaný soubor je možné ručně upravit, pokud se při exportu některé prvky třeba posunuly nebo je potřeba jim změnit velikost, nebo upravit formátování.

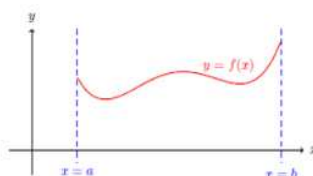
8.1 Integrovaní - „sčítání“ mnoha malých příspěvků

Studijní cíle

- Pochopit proces integrování tzv. prouzkovou metodou
- Umět vysvětlit odvození Riemannova integrálu - pomocí metody horních a dolních součtů a pomocí metody integrálních součtů

Motivace

Abychom pochopili podstatu integrování, vyjděme z formulace následujícího geometrického problému: Nechť je dána spojitá nezáporná funkce $f(x)$ v intervalu $\langle a, b \rangle$. Graf této funkce spolu s přímkami o rovnicích $x=a$, $x=b$ a osou x vymezi v kartézské soustavě souřadnic rovinný útvar. Naším úkolem je zjistit obsah tohoto rovinného obrazce.



Obr. 8.1.1

Z pohledu na [Obr. 8.1.1](#) jistě plyne, že stanovit číslo vyjadřující obsah rovinného obrazce je naprosto korektní úvaha. Z minulých školních let určitě ještě znáte nějaké vzorce pro výpočet obsahů (kruh, obdélník, lichoběžník, trojúhelník), které bychom snad mohli nějakým způsobem využít. Pokusme se nejdříve stanovit obsah obrazce alespoň přibližně. Kdybychom si nakreslili obrazec na milimetrový papír a spočítali všechny čtverečky o obsahu 1mm^2 , které se do našeho obrazce vejdou, měli bychom

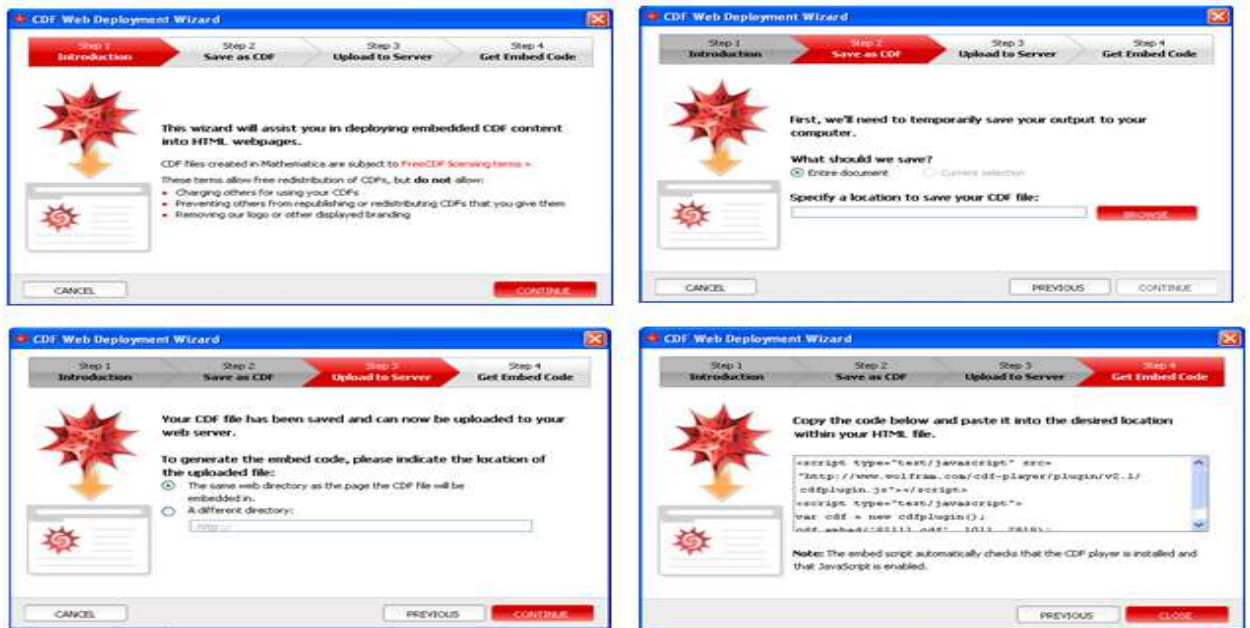
Obr. 29. Export ze software Mathematica

Existuje ještě jedna možnost jak dostat notebook do **.html*. Je možné notebook vyexportovat do formátu **.cdf*, který přehraje **CDF Player**, který se spouští pomocí javascriptového příkazu, který Mathematica vygeneruje. Průvodce exportem se spouští příkazem „**Deploy** → **Embed in HTML...**“ z menu „**File**“. Javascriptový příkaz se vloží do **.html* souboru na místo, na které chceme vložit notebook.

Export **.cdf* souboru jsem použil několikrát pro vložení přímého výstupu ze software Mathematica do stránek se vzorci renderovanými javascriptem.

Pro export celé kapitoly jsem použil výchozí nastavení exportu, tzn. vzorce jako obrázky **.gif* a vytvořit soubor CSS v automatickém umístění.

Stránka exportovaná z Mathematicy má jiné formátování než ostatní stránky. Je to proto, že některé třídy, které používám, používá i Mathematica a nemůžu proto načíst oba soubory. Při exportu všech kapitol by se vzhled upravoval CSS souborem a stránky by měli všechny jednotný vzhled.



Obr. 30. Postup exportu do *.cdf

8 POROVNÁNÍ KVALITY EXPORTU A NÁROČNOST TVORBY

I když se může zdát, že export z nějakého programu je nejméně pracný, je potřeba si uvědomit, že i zápis do onoho programu nějakou práci stál. Z hlediska pracnosti jsou si všechny systémy rovny, všechny data, které se zobrazují, musí někdo „naklepat“. Stejně tak i formátování, buď zápisem do zdrojového kódu nebo naklikáním v WYSIWYG editoru.

Dalším hlediskem může být ergonomie, tady si myslím, že taky nelze dělat rozdíly mezi klikacím editorem a editorem, do kterého se všechno zapisuje.

Nejvíce bude záležet na osobních preferencích uživatele. A tady u mne na plné čáře vyhrává zápis pomocí HTML editoru. A to ze dvou důvodů:

- Můžu ovlivnit do poslední tečky, co se bude zobrazovat, můžu si vymyslet svoje třídy a jakákoliv úprava dílem okamžiku.
- U exportu se jedná vždy o nějakou změnu původního zdroje a lehko se může stát, že se něco nevyexportuje, tak jak jsem si představoval, že se stane.

Navíc pro export musím něco dalšího instalovat do počítače, a to něco se nemusí shodnout s něčím, co již nainstalované mám. (Jedním z problémů, na které jsem při instalaci exportních programů přišel, bylo, že programu LaTeXML vadil můj antispyware. Po jeho vypnutí se instalace dostala o kus dál.)

Samozřejmě bude jiná situace pro někoho, kdo má již hromadu textů v LaTeXu a chce je dát na web. Tím ubude teoreticky pracnosti a export se dostane na špici. Pak bude záležet, jak se podaří nastavit parametry exportu, aby vše fungovalo, tak jak má.

Kvalita bude nepochybně vždy vyšší u přímého zápisu pomocí jakéhokoliv HTML editoru. Export bude vždy jenom takový pomocník, abychom nemuseli znovu přepisovat to, co už jednou napsané máme, jen někde jinde.

9 TVORBA VYBRANÉ KAPITOLY V HTML

Pro tvorbu celé kapitoly skript jsem se rozhodl použít javascriptovou knihovnu jsMath, a to z důvodů již dříve uvedených. Protože budu tvořit více stránek se vzorci, bude to datově úspornější než použití obrázků, nebude nám ani vadit nahrávání samotné knihovny. Použití jazyka MathML je z důvodu nepodpory většiny prohlížečů nepraktické. Naproti tomu jsMath funguje ve všech prohlížečích s podporou javascriptu (což jsou téměř všechny, pokud uživatel podporu explicitně nevypne). Pokud nejsou nainstalované potřebné fonty, použijí se obrázky, které jsou součástí knihovny.

9.1 Komplexní vzhled stránek

Fyzický vzhled je kompletně nastavován pomocí CSS, díky tomu je dosaženo jednotného vzhledu všech stránek. Šířku aktivního pole jsem nastavil na 800 pixelů. Je to proto, že podle statistiky má více než 60 % uživatelů internetu na svém monitoru stále rozlišení 800x600. V případě, že má vyšší rozlišení, je tato aktivní plocha na pozadí tvořeném mozaikou bitmapového obrázku.

Samotnou aktivní plochu jsem rozdělil na tři části. První je hlavička, ve které je umístěno logo fakulty. Druhou částí je tělo, které obsahuje vlastní text skript, vzorce, příklady a grafy. V poslední, třetí části, kterou jsem si nazval patičkou, jsou odkazy na ostatní podkapitoly v této kapitole, a úvodní stránku skript. V případě kompletnosti celých skript je možné přidat odkazy na hlavní stránky jednotlivých kapitol.

9.2 Tvorba samotných skript

Skripta mám k dispozici ve formátu **.pdf*. Na samotném přepisování textů není nic světoborného. Snažil jsem se používat logického formátování. Po vytvoření stránky jsem ji naformátoval pomocí CSS. Soubor s pravidly CSS jsem připojil direktivou `<link />` v hlavičce souboru. Dále ještě v hlavičce připojuji knihovny **jQuery**, její rozšíření **jQueryCycle**, můj soubor s javascriptovými skripty a knihovnu **jsMath**. Všechny tyto knihovny můžeme stahovat z jejich domovských stránek a vkládat přímé odkazy na tyto zdroje. To má tu nevýhodu, že pokud se změní zdroj (někdo přejmenuje adresář nebo soubor, vyjde nová verze), přestanou stránky fungovat. Výhodou je, že jsou stránky stále aktuální, resp. používají aktuální plug-iny.

Druhou možností je potřebné soubory uložit na svůj server a odkazovat relativně. To má tu výhodu, že skript bude fungovat stále (pokud ho nesmaže poskytovatel prostoru, pro porušení podmínek). Nevýhodou je, že když se nebude pravidelně aktualizovat, budou plug-iny zastaralé. Což nemusí úplně vadit, když fungují.

Vzorce jsou zapisovány do tagů třídy „Math“, ve kterých jsMath použije TeXovské fonty nebo svoje obrázky, spolu se svým formátováním.

Na stránkách jsou příklady, jejichž řešení je skryto a objeví se až po kliknutí na tlačítko. Toto skrývání je řešeno pomocí javascriptových funkcí, které využívají funkce knihovny jQuery. K některým řešením patří sekvence obrázků, které jsou v předloze v dalších souborech. Já je načtu pomocí plug-inu jQuery Cycle, který sadu obrázků umí vyměňovat pomocí různých animací. Tento plug-in umí asi dvě desítky různých animací. Já si vybral animaci, kdy obrázek „odletí“ pryč a na jeho místě se objeví nový.

V některých případech je použitý „pohyblivý“ obrázek. Je to tzv. animace pomocí animovaného gifu. Tzn. obrázek je ve formátu *.gif, který umí zobrazovat jednotlivé vrstvy a postupně je vystřídat. Je to efekt, který návštěvníka většinou zaujme. Není vhodné to s animovanými gify přehánět, protože s rostoucím počtem vrstev roste i velikost obrázku. A tím je rychlost načítání. Přehrávání gif, pokud je jich hodně může být zpomalený celý počítač.

ZÁVĚR

Hlavním úkolem této práce je najít nejvýhodnější cestu k vystavení studijních materiálů z Matematiky na webové stránky. Ukázat, zda je vůbec možné skripta v rozumném čase a za vyložení odpovídajícího úsilí tyto materiály vystavit.

Jak je vidět v příloze, možné to určitě je, otázkou je, který způsob je optimální. Osobně si myslím, že nejlepší je použít javascript, respektive javascriptovou knihovnu jsMath.

Obrázkové rovnice jsem zavrhl z důvodu velké datové náročnosti a obrovské pracnosti při vytváření takového kvanta obrázků.

MathML jsem nevybral z důvodu chabé podpory prohlížečů. Vlastně ze všech zkoušených prohlížečů, umí správně zobrazit vzorce zadané pouze Mozilla Firefox a prohlížeč Amaya (neznám ale nikoho, kdo by tento prohlížeč používal). Internet Explorer implicitně zobrazovat MathML neumí. Je nutné stáhnout doplněk MathPlayer z internetové adresy: <http://www.dessci.com/en/products/mathplayer/download.htm>

Export z Mathematicy dopadl nějak nemastně neslaně. Stránka vypadá obstojně, ale musí se doformátovat, nebo přeformátovat (ale to bývá složitější než celé formátování napsat znovu). Ale určitě se dá použít, ale pokud stránku mám psát celou novou, tak určitě sáhnu po HTML editoru.

Export z LaTeXu se mi nepodařil udělat v použitelné kvalitě. Pomocí programu TeX4ht se mi export částečně podařil, ale musíme zapsat zdrojový text úplně stejně, ale stránku psanou v jsMath, můžeme více upravit.

ZÁVĚR V ANGLIČTINĚ

The main goal of this work is to find the most favourable way of putting the educational materials from Mathematics on the websites and to show if it is possible to display the materials in a reasonable time and put corresponding effort into displaying them at all.

As it can be seen, it certainly is possible. The question is, what method is optimal. In my opinion, the best way to do it, is by javascript, or javascript library jsMath.

I rejected the method of pictures with equations because of the huge data demandingness and enormous arduousness to create such amount of pictures.

I did not choose MathML because of poor support of the browsers. Mozilla Firefox and Amaya (although I do not know anybody using this one) were the only two browsers capable of portraying the given formulas properly. The Internet Explorer cannot portray the MathML implicitly. It is necessary to download the accessory MathPlayer from the following website: <http://www.dessci.com/en/products/mathplayer/download.htm>

Exports fell from Mathematica somehow uninspired. This page looks fairly good, but it must endformat or reformat (but it is more complicated than all formatting write again). But surely we can use, but if I write a page a new one, so certainly I touch the HTML editor.

Export LaTeX I could not do in a usable quality. Using export TeX4ht I partially succeeded, but we have to write exactly the same source text, but written in jsMath page, you can edit more.

SEZNAM POUŽITÉ LITERATURY

- [1] OSTRAVSKÝ, Jan a Vladimír POLÁŠEK. *Diferenciální a integrální počet funkce jedné proměnné*. Zlín: Univerzita Tomáše Bati ve Zlíně, 2011. ISBN 978-80-7454-124-7. Dostupné z:
https://web.fai.utb.cz/cs/docs/Skripta_Matematika_1_2011.zip.
- [2] KOPKA, Helmut a Patrick W DALY. *Latex: podrobný průvodce*. Vyd. 1. Překlad Jan Gregor. Brno: Computer Press, 2004, vii, 576 s. ISBN 80-722-6973-9.
- [3] RYBIČKA, Jiří. *Latex pro začátečníky*. Brno: Konvoj, 2003, 238 s. ISBN 80-730-2049-1.
- [4] WOLFRAM, Stephen. *The Mathematica book*. 5. ed. Champaign, Ill: Wolfram Media, 2003. ISBN 15-795-5022-3.
- [5] *Tvorba-webu.cz* [online]. © 2003-2008 [cit. 2012-06-07]. Dostupné z:
www.tvorba-webu.cz
- [6] M. GURARI, Eitan. *TEX4ht: HTML Production* [online]. 2004 [cit. 2012-06-07]. Dostupné z: <http://www.tug.org/TUGboat/tb25-1/gurari.pdf>
- [7] *Wolfram Research: Mathematica, Technical and Scientific Software* [online]. ©2012 [cit. 2012-06-07]. Dostupné z: www.wolfram.com
- [8] *TRIAL: Matematická analýza* [online]. 2012, 1.6. [cit. 2012-06-07]. Dostupné z: <http://trial.zcu.cz/?page=predmet&ob=MA1&in=MA1&volba=w&zmena=1>
- [9] *Matematika II* [online]. [cit. 2012-06-07]. Dostupné z:
<http://homen.vsb.cz/~kre40/esfmat2/>
- [10] *Informace pro studenty AMA1* [online]. [cit. 2012-06-07]. Dostupné z:
<http://www.umat.feec.vutbr.cz/~krupkova/ama1n.html>
- [11] CHRAMCOV, Bronislav. *Základy práce v prostředí Mathematica*. Vyd. 2. Ve Zlíně: Univerzita Tomáše Bati, 2006, 122 s. ISBN 80-731-8510-5.
- [12] *JsMath: A Method of Including Mathematics in Web Pages* [online]. 2004, 10.3.2009 [cit. 2012-06-07]. Dostupné z:
<http://www.math.union.edu/~dpvc/jsmath/>

-
- [13] *World Wide Web Consortium: W3C* [online]. © 2012 [cit. 2012-06-07]. Dostupné z: <http://www.math.union.edu/~dpvc/jsmath/>
- [14] *LaTeX2HTML* [online]. 2001 [cit. 2012-06-07]. Dostupné z: <http://www.latex2html.org>
- [15] *LaTeXML: A LaTeX to XML Converter* [online]. 2012, March 26 [cit. 2012-06-07]. Dostupné z: <http://dlmf.nist.gov/LaTeXML>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

Apod.	A podobně
ASCII	American Standard Code for Information Interchange – americký standardní kód pro výměnu informací
CERN	Conseil Européen pour la Recherche Nucléaire – evropská organizace pro jaderný výzkum
FAI	Fakulta aplikované informatiky
GUI	Graphical User Interface – grafické uživatelské rozhraní
HTML	HyperText Markup Language – hypertextový značkovací jazyk
MathML	Mathematical Markup Language – matematický značkovací jazyk
MIT	Massachusetts Institute of Technology - Massachusettský technologický institut
Např.	Například
PC	Personal Computer – osobní počítač
Tzv.	Takzvaný
VŠB-TU	Vysoká škola báňská – Technická univerzita
VUT	Vysoké učení technické
WYSIWYG	What you see is what you get – co vidíš, to dostaneš
W3C	World Wide Web Consortium
XHTML	Extensible HyperText Markup Language – rozšířitelný hypertextový značkovací jazyk
XML	Extensible Markup Language – rozšířitelný značkovací jazyk
ZČU	Západočeská univerzita

SEZNAM OBRÁZKŮ

Obr. 1. Stránka materiálů VŠB-TU Ostrava	12
Obr. 2. Stránka materiálů VUT v Brně	13
Obr. 3. Stránka materiálů ZČU v Plzni	14
Obr. 4. Editor TeXnicCenter	17
Obr. 5. Možnost exportu z LaTeXu v editoru TeXnicCenter	20
Obr. 6. Logo Wolfram Mathematica 8 (Spikey)	27
Obr. 7. Buňky software Mathematica	28
Obr. 8. Palety v software Mathematica (Basic Math Input a Basic Math Assistant)	28
Obr. 9 Hlavní menu software Mathematica	29
Obr. 10. Definice uživatelské funkce	30
Obr. 11. Výpis všech parametrů funkce Plot	32
Obr. 12. Interaktivní nápověda v software Mathematica	34
Obr. 13. Okno editoru se zvýrazňováním HTML kódu	36
Obr. 14. Vložení intervalu pomocí standartních znaků ASCII	43
Obr. 15. Vložení intervalu pomocí obrázku	43
Obr. 16. Vložení rovnic jako obrázků	44
Obr. 17. Vložený obrázek na nebílém pozadí	44
Obr. 18. Vzorec vykreslený pomocí jsMath	45
Obr. 19. Vložený vzorec na nebílém pozadí	46
Obr. 20 Internet Explorer tagy jazyk MathML úplně ignoruje	47
Obr. 21. Chrome (jádro Webkit) se snaží značky interpretovat	47
Obr. 22. Zobrazení v experimentálním prohlížeči Amaya	47
Obr. 23. Zobrazení v prohlížeči Mozilla (jádro Gecko)	47
Obr. 24. Vložení vzorce na barevné pozadí	48
Obr. 25. Export z LaTeXu pomocí TeX4ht	50
Obr. 26. Wolfram's Mathematica – Save As	51
Obr. 27. Volba formátu exportu	51
Obr. 28. Možnosti exportu do html Wolfram's Mathematica	52
Obr. 29. Export ze software Mathematica	53
Obr. 30. Postup exportu do *.cdf	54

SEZNAM PŘÍLOH

- A. CD-ROM s elektronickou verzí BP a elektronickými materiály ve formátu *.html