Jihočeská univerzita v Českých Budějovicích
PEDAGOGICKÁ FAKULTA
KATEDRA INFORMATIKY

MULTIMEDIÁLNÍ MATERIÁL PRO PRAKTIKA Z FYZIKY

Bakalářská práce

vedoucí práce
Ing. Michal Šerý

Zdeněk Česák

České Budějovice, červen 2008

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské, a to v nezkrácené podobě - v úpravě vzniklé vypuštěním vyznačených částí archivovaných pedagogickou fakultou elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách.

V Českých Budějovicích 2008 _____

Anotace

Předmětem této práce je získání, úprava a konečná prezentace souboru materiálů pro praktika z fyziky. Její součástí je tvorba interaktivních stránek a prezentace přes www rozhraní a jejich implementace do systému eAmos.

Annotation

Subject of this work is obtaining of set of materials for practices from physics and its arrangement and final presentation. The part of this work is also creation of interactive sites and presentation over www borderline. Another part is implementation of these sites to system eAmos.

Poděkování

Děkuji panu Ing. Michalovi Šerému, vedoucímu mé bakalářské práce, za jeho odborné vedení, cenné rady a připomínky, kterými mi pomohl při jejím vypracování.

Mé poděkování patří také Janu Prollovi za poskytnuté materiály k jednotlivým fyzikálním měřením, jejich setřídění, pořízení fotografií a videí.

Obsah

1 ÚVOD

Podnětem ke vzniku této práce byla potřeba řady studentů dálkového studia MVT na JČU nějakým způsobem získat podklady ke cvičením. Při některých fyzikálních měřeních jsme naráželi na nedostatek jakýchkoli materiálů pro zpracování odevzdávaných protokolů.

Celý projekt je rozdělen do dvou bakalářských prací. První práce je zaměřena na sebrání a setřídění jednotlivých měření z praktik pro jeden semestr. V našem případě se jedná o předmět Základy fyzikálních měření III, vyučovaném v zimním semestru druhého ročníku. Tuto část zpracovává jako bakalářskou práci Jan Proll. Druhá část projektu má za úkol tato data upravit a včlenit do interaktivní www prezentace a integrovat do systému eAmos. Toto téma jsem si pro zpracování zvolil já a to proto, abych přiblížil danou tématiku ostatním studentům a ulehčil tak jejich příští studium, aby se nemuseli potýkat s problémy a nedostatkem materiálů, jako další absolventi našeho oboru.

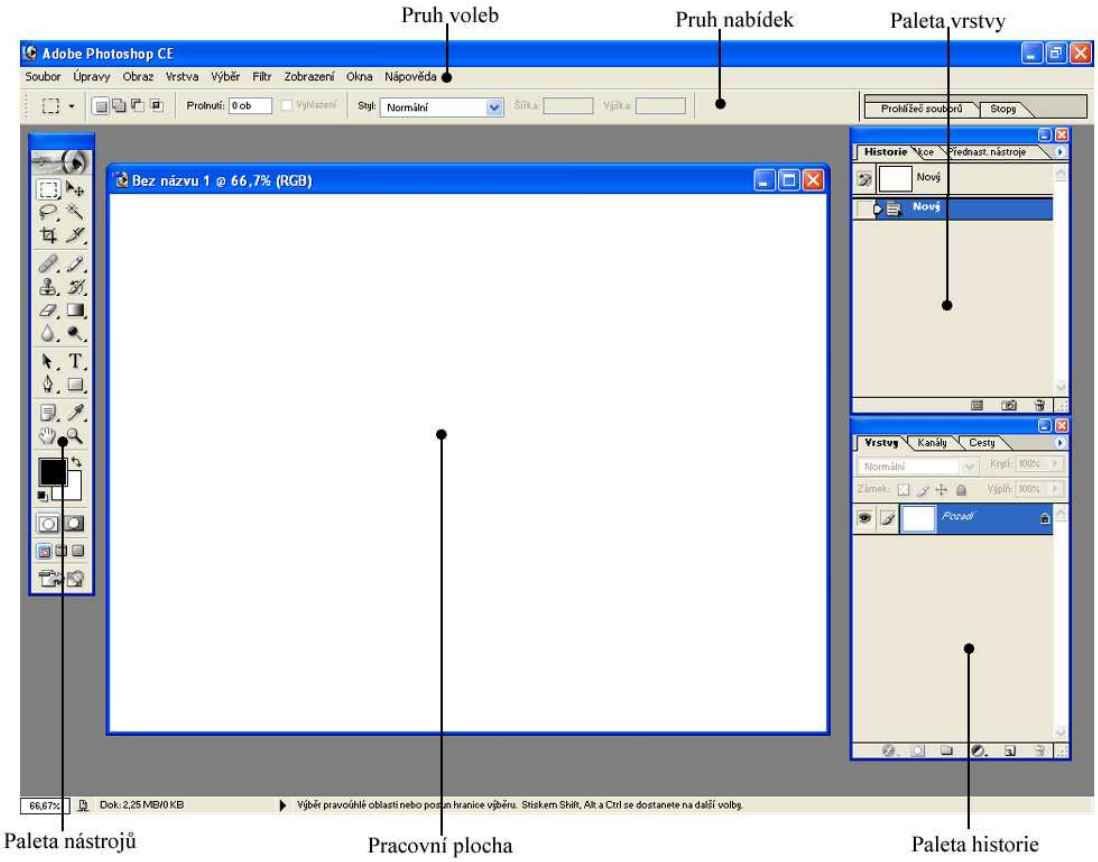
2 ÚPRAVA MATERIÁLŮ

2.1 Úprava fotografií – ADOBE PHOTOSHOP

Pro úpravu fotografií jsme si vybrali grafický editor Adobe Photoshop 7.0 cz. Tento editor jsme zvolili z důvodu poměrně „snadného“ a intuitivního ovládání v českém jazyce a s ohledem k určitým předchozím zkušenostem práce s tímto programem.

2.1.1 Popis programu

Pohled na pracovní plochu



obr. č. 1

2.1.2 Některé nástroje použité k úpravě fotografií

Nástroj Obdélníkový výběr  **:**

Dokáže vytvořit obdélníkový výběr (v podnabídce nástroje lze najít i jiné tvary) a to klepnutím a táhnutím myši. První klepnutí tvoří první roh výběru, a místo, kde uvolníte tlačítko myši, vytvoří protilehlý roh. Pokud chceme vybrat čtvercovou oblast, přidržíme klávesu Shift a poté začneme kreslit výběr. Pro variantu kreslení výběru od středu směrem k vnější hraně přidržíme Alt. Tyto možnosti lze kombinovat.

Nástroj Oříznutí  **:**

Tento nástroj se používá podobně jako předchozí- Obdélníkový výběr. Nevytváří selekci, ale umožňuje oddělit nechtěné části obrázku. S využitím tohoto nástroje můžete obrázek zároveň oříznout, změnit jeho velikost a také jej otočit. Umožňuje také změnu perspektivy, která ale pro naši práci nebude potřebná. Provedení „ořezu“ aplikujeme klávesou Enter.

Nástroj Lupa  **:**

Po klepnutí na obrázek nástrojem Lupa zvětšíte (zmenšíte) zobrazení na předem definovanou úroveň (implicitně 25 %, 33,33 %, 50 %, 66,67 %, 100 %, 200 %, 300 % a podobně).

2.1.3 Vlastní úprava fotografií

Množství úprav na fotografiích našeho typu nebylo příliš vysoké. V podstatě se jednalo pouze o ořezávání, vytváření náhledů a v některých případech o doostřování určitých detailů. Ořezávání fotografií je popsáno v předchozí

kapitole a provází se pouze nástrojem ořízнutí. Při vytváření náhledů do galerie bylo použito nástroje: ***velikost obrazu***. Tato možnost se nachází v pruhu voleb, v možnosti obraz. Při změnění velikosti fotografie je důležité mít zatrženo volbu: ***zachovat proporce***. Poté se při změně jednoho rozměru obrazu automaticky přepočítá i druhý, tímto zajistíme, že změněná fotografie si zachová stejný poměr stran. Pokud je objekt vkládaný do prezentace příliš velký a hrozí,v případě pomalého připojení k internetu jeho dlouhé stahování, máme tři možnosti jak daný obraz zmenšit. První je již zmiňovaná změna velikosti obrazu, druhá je zmenšení jeho rozlišení (DPI z anglického jazyka „Dots per inch“, což je hodnota, která udává kolik obrazových bodů (pixelů) se vejde do délky jednoho palce.) Tato volba se nachází také v možnosti velikost obrazu. Třetí možností je volba vhodného formátu fotografie (jpg, gif, bmp atd.) a její komprese. Photoshop umožnuje pro tento případ použití volby: ***Uložit pro web***.

2.1.4 Doostřování detailů

Je-li rozostřena pouze nějaká část fotografie, ať z důvodu různé hloubky ostrosti nebo jinou chybou při fotografování, můžeme výsledek částečně napravit podle následujícího postupu. Poškozenou část fotografie označíme například: ***obdélníkovým výběrem***. Klikneme na vybranou část pravým tlačítkem myši a vybereme volbu: ***vrstva vyjmutím***. Tímto si oddělíme pouze poškozenou část a zbytek fotografie již upravovat nebudeme. V pruhu voleb vybereme položku: ***filtr***, dále zostřit a poslední vybere možnost: ***doostřit***. Tento filtr je jediný z nabídky neautomatický, nastavuje se manuálně a tudíž je vždy vidět co se s fotografií přesně děje. Po kliknutí na tuto možnost se otevře dialogové okno s nastaveními filtru, kterými můžeme volit parametry ostření obrázku. K nastavování parametrů slouží tři posuvníky. Posuvníkem: ***míra*** nastavujeme sílu účinku ostření. Prostředním s označením: ***poloměr***,

nastavujeme velikost oblasti v níž bude filtr porovnávat pixely a ostřit. Dá se říci, že čím větší oblast nastavíme, tím dramatičtější budou změny. Poslední posuvník s označením *práh* nastavuje úroveň, nad kterou se bude ostřící účinek projevovat. Čísla 0 až 255 v podstatě odpovídají hodnotám jasu. Pokud je jasový rozdíl větší než nastavený práh, dojde v daném místě k zostření obrázku. Pokud je menší ke změnám nedojde.

3 Prostředky použité k tvorbě prezentace

Celá prezentace je z důvodu jednoduchosti tvořena v HTML kódu, do kterého jsou vkládány části kódu v jazyce PHP a JavaScripty. Jazyk PHP je používán v místech, kde je třeba kód nějakým způsobem dělit na několik částí a v HTML kódu by struktura byla nepřehledná a složitá. JavaScript zajišťuje práci s okny a fotografiemi, protože je vhodnější než PHP.

3.1 Struktura a vlastnosti HTML kódu

Zkratka HTML znamená „Hypertext Markup Language“. Jedná se o jazyk pro tvorbu dokumentů, který definuje celkový vzhled textu (velikost nadpisů, použité fonty písma, okraje, atd.). Jazyk HTML byl speciálně vyvinut (a stále se vyvíjí) za účelem publikování dokumentů na www.

Zdrojový kód dokumentu psaném v jazyce HTML, je text psaný v ASCII formátu, který lze prohlížet i upravovat v libovolném textovém editoru. HTML soubor je jako celek předán ze serveru prohlížeči, který danou www stránku zavolal. Ten funguje jako překladač a zobrazí uživateli již hotovou stránku. Tomuto způsobu předávání informací se říká „programování na straně uživatele“. Opačný způsob využívá například programovací jazyk PHP. Tento způsob se nazývá „programování na straně serveru“ a funguje tak, že server přeloží uživatelem volaný soubor do kódu html a tento výsledný text pošle prohlížeči uživatele. Tento způsob má nespornou výhodu v tom, že se uživatel nedostane ke zdrojovým textům. Výhodou prvního způsobu je schopnost dynamicky reagovat na událost způsobenou klientem (např. pohyb kurzoru myši, apod.), což PHP nedokáže, protože k provedení každé své nové události, musí být požadavek vždy prohlížečem znovu odeslán na server. Proto je nejvhodnější variantou kombinovat PHP s JavaScriptem nebo jiným, dynamicky reagujícím jazykem.

Každý správně formátovaný HTML dokument by měl na začátku obsahovat informace o verzi použitého HTML a typu DTD (Document Type Definition). DTD je jinými slovy návod pro prohlížeč zpracovávající dokument. Říká mu, jaké elementy dokument používá a jak s nimi zacházet. Element DOCTYPE není součástí HTML dokumentu, není elementem HTML a nemusí mít koncovou značku. Prohlížeče rozlišují dva základní vykreslovací režimy, jimž se také říká módy.

3.1.1 !DOCTYPE

3.1.1.1 Standartní mód

Jde o blokový model vykreslování, jenž se chová podle specifikace W3C. Používají je Mozilla (+ odvozeniny), Internet Explorer 6 a 7 (se striktním doctype) a Opera (se striktním doctype).

Začátek kódu stránky se striktním doctype vypadá třeba takto:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01">
<html>...
```

nebo

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

3.1.1.2 Nestandartní QUIRK mód

Také se tomu říká "režim zpětné kompatibility" nebo "kompatibilní režim". Používají ho všechny verze Internet Exploreru (kromě šestky a sedmičky se striktním doctype) a Opera bez striktního doctype.

Příklad začátku stránky vykreslovaný v Exploreru v quirk módu:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
...
</html>
```

nebo bez doctype:

```
<html>
...
</html>
```

- **Rozdíly mezi módy:**

Mezi těmito dvěma módy jsou nejdůležitější tři rozdíly v zobrazování stránek:

jinak počítají šířku a výšku (boxmodel)
jinak zobrazují velikost písma (quirk mód má písmo o stupeň větší)
quirk mód dovoluje nastavovat rozměry řádkovým prvkům

3.1.2 VLASTNÍ HTML KÓD

Až na několik vyjímek (např
) se HTML kód zapisuje v párových tagech. Zde je několik příkladů a ukázka základní struktury HTML kódu:

```
<html>
  <head>
    <title>Stránka</title>
  </head>

  <body>
    Vlastní kód stránky.
  </body>
</html>
```

- **Co znamenají jednotlivé tagy:**

<**html**> </**html**> začíná a končí dokument
<**head**> </**head**> začíná a končí hlavičku, která se sice nezobrazuje, ale obsahuje některé důležité údaje, například
<**title**> </**title**> vymezující název dokumentu (může se lišit od jména souboru)
<**body**></**body**> co je mezi těmito dvěma tagy, se bude zobrazovat.

Toto je příklad základní struktury dokumentu. Výše uvedené tagy by měl obsahovat každý HTML soubor.

3.2 Jazyk PHP

Jak již bylo zmíněno dříve, jazyk PHP řadíme do skupiny skriptovacích jazyků, které se provádějí na straně serveru. PHP je na serveru závislé, protože na něm běží jeho interpreter, který skripty provádí. PHP se tímto odlišuje např. od JavaScriptu, jehož skripty se stahují přímo s HTML stránkou a jsou vykonány na straně klienta jeho prohlížečem.

Příklad:

Takto vypadá kód HTML obsahující PHP script

```
<HTML>
<HEAD>
<TITLE>Příklad</TITLE>
</HEAD>
<BODY>
Dnes je <?echo Date("w. m. Y");?>
</BODY>
</HTML>
```

A takto vypadá výsledný zdrojový kód na straně klienta

```
<HTML>
<HEAD>
<TITLE>Příklad</TITLE>
</HEAD>
<BODY>
Dnes je 4. 05. 2000
```

```
</BODY>
</HTML>
```

3.2.1 Vkládání scriptů do HTML a syntaxe jazyka

Zdrojový kód PHP skriptu se vkládá přímo do zdrojového kódu HTML stránky. Z toho vyplývá, že tak jako v HTML má každý element svůj začátek a konec, mezi kterým je definován, tak i PHP skript musí být definován mezi svými HTML TAGy, aby server poznal, kde skript začíná a kde končí. Způsobů, jak PHP skript vložit do HTML stránky, je více. Dva nejpoužívanější jsou dvojice elementů

```
<? echo “Zde vložíme script“;  ?>
```

anebo dvojice jim velmi podobná

```
<?php “Zde vložíme script“; ?>
```

Třetí způsob se podobá vkládání JavaScriptů do HTML stránek, ale je trochu zdlouhavější. PHP skript se v tomto případě zapisuje takto:

```
<HTML>
<HEAD>
<TITLE>Vkládání PHP do HTML - 3. způsob</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="php">
   echo "Třetí způsob";
</SCRIPT>
```

```
</BODY>
</HTML>
```

Syntaxe PHP se velice podobá syntaxi programovacího jazyka C, ze kterého původně jazyk PHP vyšel. Každý příkaz musí být oddělen od příkazu dalšího. Příkazy od sebe můžeme oddělit středníkem nebo každý příkaz uvést samostatně mezi TAGy.

První způsob - pomocí středníků:

```
<? echo "Tohle je kratší"; echo "a lépe se v tom orientuje"; ?>
```

Druhý způsob - pomocí TAGů:

```
<? echo "Tohle je to samé" ?><? echo "ale delší" ?>
```

3.2.2 Proměnné v PHP

Na rozdíl od některých jiných programovacích jazyků, není v PHP třeba deklarovat proměnné předem. K jejich zápisu se používá ***znak $***, za ním je název proměnné. Obecně by se dalo říci, že proměnné slouží k uchování hodnot v rámci skriptu. Může v nich být uloženo cokoliv - PHP si samo určí typ hodnoty.
Typy hodnot:

INTEGER - hodnota je celé číslo v rozsahu od -2 147 483 648 do 2 147 483 647

DOUBLE - hodnota je desetinné číslo v rozsahu od $-1{,}7 \times 10^{308}$ do $1{,}7 \times 10^{308}$. Pokud chceme do proměnné uložit číslo v exponenciálním tvaru, zápis vypadá takto: `$a = 7.4e5;` ($= 7{,}4 \times 10^{5}$).

STRING - hodnotou je znakový řetězec. Ten se zapisuje mezi uvozovky nebo apostrofy.

ARRAY - proměnná, která obsahuje více hodnot. Česky tento typ proměnné můžeme nazvat pole. Každou hodnotu proměnné vyvoláváme pomocí indexu prvku pole, kterou zapisuje do hranatých závorek hned za proměnnou.

3.2.3 Větvení scriptu, cykly

Větvení scriptu se využívá v případech, kdy je za různých podmínek potřeba vykonat různé bloky kódu PHP umožňuje dvě podmíněné konstrukce. První je ***if … else***, který nám umožňuje porovnávat různé výrazy a v závislosti na výsledku jejich porovnání, vykonat jisté příkazy nebo bloky příkazů. V případě, že potřebujeme porovnávat místo více hodnot jediný výraz, PHP umožňuje konstrukci ***switch … case***, která tuto operaci umožnuje.

3.2.3.1 Příkaz if

Příkaz if je jedním z nejdůležitějších příkazů nejen v jazyce PHP, ale pravděpodobně ve všech programovacích jazycích. Umožňuje provádět vybrané části kódu v závislosti na tom, jaké nastanou dané podmínky. Syntaxe příkazu je jednoduchá – *if (podmínka) příkaz;*. Pokud se má provést sekvence

více než jednoho příkazu, zapisují se mezi složené závorky {}. Příklad z prezentace, který zajišťuje správné vypisovaní menu:

```
<?php
if ($menu=="2")
        { echo "<br>
                <a href = 'main.php?co=sites/r2zs.php&menu=2'>ZS</a>
                <br>
                <a href = 'main.php?co=error.htm&menu=2'>LS</a>
                <br><br>";
        }
?>
```

Příkazy ve složených závorkách se provedou pouze v případě, že výsledek porovnání dané podmínky je rovno hodnotě “true” (pravda). Pokud výsledkem bude hodnota “false“ (nepravda), příkaz mezi závorkami se přeskočí a vůbec se neprovede. Pokud je potřeba provézt příkaz v tomto případě, umožňuje nám to PHP pomocí klíčového slova ***else***. Každý z těchto podmíněně prováděných bloků se nazývá **větev** a každá větev musí, v případě, že obsahuje více než jeden řádek kódu, být umístěna uvnitř složených závorek.

Příklad:

```
<?php
if ($cislo<0)
        {echo ‘Cislo ’.$cislo.’ Je zaporne.’;
        echo ’<br>’
        }
 else {echo ’Cislo je zaporne.’;
        }
```

```
?>
```

PHP obsahuje také klíčové slovo ***elseif***, které umožňuje v případě nesplnění hlavní podmínky porovnávat alternativní podmínky. Za jedním příkazem ***if*** může následovat několik dalších příkazů ***elseif***. Poslední větev ***else*** umožňuje vložit kód, který má být proveden za předpokladu, že nebyla splněna ani jedna z předcházejících podmínek za ***if*** nebo ***esleif***.

3.2.3.2 Příkaz switch

Příkaz ***switch*** slouží k porovnání jedné proměnné s větším množstvím argumentů. V tomto případě je nejen zbytečné, ale hlavně poměrně nepřehledné, vypisovat za sebe větší množství příkazů ***if … elseif …elseif*** atd. Syntaxe vypadá takto:

```
switch ($promenna) {
        case ”podminka1”:  prikaz1;
        break;
        case “podminka2“: prikaz2;
        break;
        default: prikaz3;
                }
```

V tomto případě se postupně porovnává ***$promenna*** s hodnotou ***podminka1***, pak ***podminka2***. Pokud ani jedna z nich není splněna, provede se ***příkaz3*** za klíčovým slovem ***default***. Příkaz ***break*** ukončuje sekvenci příkazů, které mají být v případě, že je podmínka splněna, vykonány.

3.2.3.3 Cykly

Cykly slouží v programování k opakování určitého bloku kódu vícekrát za sebou. Cykly můžeme rozdělit na dva základní typy. První je cyklus s proměnným počtem kroků. Tento typ je v PHP prezentován příkazem **w*hile***. Druhým typem je cyklus s přesným počtem kroků. Tento je prezentován příkazem ***for***.

3.2.3.4 Cykly while

Cyklus ***while*** je nejjednodušším příkazem cyklu. Cyklus ***while*** se dá použít dvěma způsoby a to jako cyklus s podmínkou na začátku nebo na konci cyklu. Syntaxe je podobná příkazu ***if***.

```
while (podmínka) { příkazy
        }
```

Tento způsob zápisu je typ cyklu s podmínkou na začátku. V prvním kroku se vyhodnocuje podmínka uvedená mezi kulatými závorkami. Pokud je tento výraz vyhodnocen jako pravdivý, provádí se kód uvnitř složených závorek. Když program dojde k uzavírající závorce }, je podmínka opakování vyhodnocena znovu a pokud je stále vyhodnocená jako pravda, provede se blok příkazů uvnitř složených závorek znova. Toto se opakuje do té doby, dokud není podmínka vyhodnocena jako nepravdivá. Tato podmínka je testována pouze a počátku každé iterace, takže pokud se změní někde uprostřed cyklu, kód bude proveden až do konce. Přerušení kódu je možné pomocí příkazu ***break***. Cyklus ***while*** se dá psát i jiným způsobem a to pomocí příkazu

do … while. Tento způsob je velmi podobný předchozímu příkazu ***while***, jen s tím rozdílem, že podmínka je testována na konci každého cyklu místo na začátku. To znamená, že cyklus se provede vždy alespoň jednou. Zápis vypadá takto:

```
do { blok příkazů
   }
while (podmínka)
```

Proměnné, které se používají v příkazech ***while*** a ***do …while*** v řídícím výrazu, se někdy nazývají **řídícími proměnnými cyklu**. Příkazy ***while*** se obykle používají ke čtení z databázových dotazů, řádků ze souboru nebo položek z pole.

3.2.3.5 Cykly for

Syntaxe cyklu for je trochu složitější, ačkoliv jsou často vhodnější, než u cyklů *while.*

```
for ($i = 1; $i < 11; ++$i)
        {
        echo ("$i <BR>") //vypíše čísla od 1 do 10
         }
```

Do závorek za klíčovým slovem ***for*** se zadávají tři výrazy oddělené středníky. Prvním je výraz přiřazení pro inicializaci řídící proměnné cyklu. Toto přiřazení se provede pouze jednou, ještě před první iterací cyklu. Druhým výrazem je podmínka cyklu, která se vyhodnocuje na začátku každého cyklu.Pokud je tento výraz vyhodnocen jako true, provede se blok příkazů

napsaný mezi složené závorky. V opačném případě cyklus skončí. Třetí je příkaz, který se provede na konci každé iterace. Obvykle se používá pro inkrementaci nebo dekrementaci řídící proměnné cyklu.

3.3 JavaScript

JavaScript je programovací jazyk, který se zapisuje přímo do HTML kódu. JavaScript se odesílá spolu s HTML stránkou do prohlížeče klienta. V tomto je nevýhoda, že scripty může vidět kdokoliv, kdo zavítá na danou stránku. Skript se zapisuje do HTML mezi párové tagy

```
<script>
….
</script>.
```

Všechno, co je mezi těmi tagy, je program psaný v jazyce Javascript. Pokud se zapisuje normální text, musí se psát mezi uvozovkami. Každý příkaz JavaScriptu se ukončuje středníkem.

```
<script>
document.write("A toto napsal JavaScript");
</script>
```

3.3.1 Funkce v JavaScriptu

Funkce se zapisují také přímo do HTML kódu. Jejich deklarace se provádí klíčovým slovem function. Syntaxe zapisu je takováhle:

```
function jmenoFunkce(parametry) {tělo funkce};
```

nebo podrobněji zapsáno:

```
function jmenoFunkce(parametr, parametr)
{
příkaz; příkaz; return hodnota
};
```

Příklad použitý v prezentaci:

```
<SCRIPT LANGUAGE="JavaScript">
<!--//
function openWindow(x,y) {
nahled=window.open("","fotky",'toolbar=0,location=0,scrollbars=0,
resizable=0,top=20,left=20');

with (nahled) {
                        close (onClick); }

}
//-->
</SCRIPT>
```

Tato funkce slouží k otevírání samostatného okna, do kterého se zobrazí obrázek nebo fotografie. V našem případě je užita v sekci galerie.
Volání funkce se provádí pomocí jejího jména a uvedením všech nutných parametrů. Velmi často se funkce volají na základě událostí dokumentu přímo z HTML kódu, například:

```
<a href="stranka" onclick="openWindow(parametry);" ><img
src="nahleh.jpg"></a>
```

Při kliknutí na obrázek ***nahled.jpg*** se vyvolá funkce ***openWindow(),*** která v našem případě otevře samostatné okno s hodnotou parametrů. Předtím samozřejmě musí být funkce inicializovaná.

4 WWW STRÁNKY

Z důvodu toho, že v době, kdy vznikala tato práce, nebyl soubor prezentovaných dat na www stránkách úplný, vznikly www stránky pouze ukázkově pro soubor úloh měření ze zimního semestru druhého ročníku studia orientované na měření z optiky. Konkrétně se jedná o tyto úlohy:

Studium stojatého vlnění struny
Mikroskopická měření
Optické laboratorní metody
Studium parametrů dalekohledu
Charakteristiky fotocitlivých prvků
Měření ohniskové vzdálenosti tenkých čoček
Studium tlumených kmitů
Ohybové jevy na optické mřížce
Akustická měření
Index lomu na hranolu
Základní fotometrická měření

Měření ze zbývajících předmětů budou později zpracována a doplněna.

Celý text předlohy měření **Studium tlumených kmitů** viz. příloha 1.

4.1 Struktura stránek

Struktura prezentace byla, z důvodu snahy o maximální možnou přehlednost a jednoduchost, rozdělena na dvě části: programovou a datovou.

Snahou, při navrhování struktury prezentace, bylo maximální zjednodušení jak administrátorské části prezentace, tak uživatelské.

- **Datová část**

Zde jsou zahrnuty materiály týkající se jednotlivých měření. Vlastní dokumenty jsou pro přehlednost rozděleny do adresářů po jednotlivých semestrech, ve kterých se daný předmět vyučuje. Jako formát jednotlivých cvičení byl, z důvodu úspory místa a rychlejšího načítání stránek, zvolen formát ***pdf***. Tato část prezentace také obsahuje některé fotografie a obrázky pro doplnění daného předmětu a pro detailnější dokreslení požadavků na studenta, ku příkladu schémata elektrických obvodů, konkrétní přístroje používané při měření apod. Snahou je maximální zmenšení velikosti všech materiálů.

- **Programová část**

Ta obsahuje PHP kódy, HTML kódy, JavaScripty, kaskádní styly a obrázky dotvářející celkový vzhled celé prezentace. Dalším popisem této části se budeme zabývat v následující kapitole.

4.1.1 Stručný popis programové části

Jako první se spouští skript ***index.php***. Ten pouze přiřadí hodnoty do pomocných proměnných ***$menu*** a ***$co***, tyto proměnné nesou informaci o tom jaká stránka a jaké menu se má v následujícím skriptu zobrazit. Poté se zavolá script ***main.php***. Tato část kódu určuje základní vzhled celé prezentace. Párovými tagy

```
<div class="">
…..
</div>
```

v kombinaci s kaskádními styly, které jsou uloženy v souboru ***style.css***, se okno prohlížeče rozdělí na dvě vodorovné části. V horní je logo Jihočeské univerzity a katedry fyziky. Spodní pruh je dále rozdělen na levou a pravou část. Do levé je volán skript ***menu.php***, obsah pravé je určen hodnotou proměnné ***$co***. Toto uspořádání stránek má výhodu v tom, že se prezentace chová jako při použití rámců. Při kliknutí na jakýkoliv odkaz je stále volán script ***main.php***, ale jeho obsah se neustále mění podle hodnot proměnných ***$menu*** a ***$co***, které jsou přiřazovány při kliknutí na odkaz. Všechny zbývající části prezentace jsou uloženy z důvodu přehlednosti v samostatných adresářích:

- **Sites** - zde jsou všechny scripty dotvářející celou prezentaci
- **Images** - fotky a obrázky netýkající se fyzikálních materiálů a galerie
- **data**: dále rozdělen na ročníky a semestry, zde jsou všechny materiály k měřením

Poslední důležitý script programové části je ***menu.php***. Tato část kódu rozbaluje aktuální část menu, podle hodnoty uložené v proměnné ***$menu***. Tato nabývá číselných hodnot podle ročníku, který chceme zobrazit.

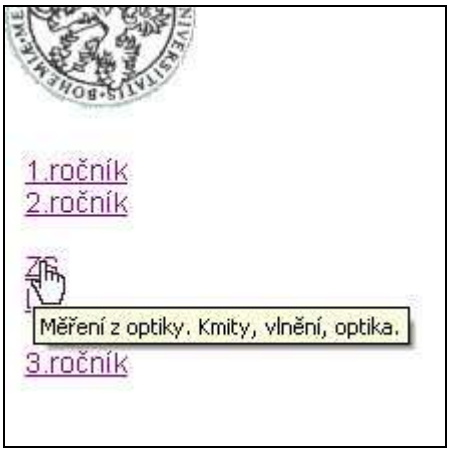
Zdrojový kód toho scriptu:

```
<a href = "main.php?menu=1&co=sites/uvod.php" >1.ročník</a><br>
```

```
<?php
    if ($menu=="1")
        { echo "<br>
         <a href = 'main.php?co=error.htm&menu=1'>ZS</a>
         <br>
             <a href = 'main.php?co=error.htm&menu=1'>LS</a>
             <br><br>";
            }
    ?>


    <a href = "main.php?menu=2&co=sites/uvod.php" title='Měření z optiky a
    elektřiny'> 2.ročník </a><br>

    <?php
   if ($menu=="2")
        { echo "<br>
          <a href = 'main.php?co=sites/r2zs.php&menu=2'title='Měření z optiky.
    Kmity, vlnění, optika.'>ZS</a>
         <br>
         <a href = 'main.php?co=error.htm&menu=2'>LS</a>
         <br><br>";
         }
    ?>

    <a href = "main.php?menu=3&co=sites/uvod.php" > 3.ročník </a><br>


    <?php
       if ($menu=="3")
```

```
        { echo "<br>
          <a href = 'main.php?co=error.htm&menu=3'>ZS</a>
           <br>
          <a href = 'main.php?co=error.htm&menu=3'>LS</a>
          <br><br>";
         }
?>
```

Odkazy v menu jsou pro snazší orientaci v prezentaci upraveny tak, aby se po najetí myši na daný předmět či měření, hned zobrazily informace týkající se tohoto předmětu viz. obrázek číslo 2.



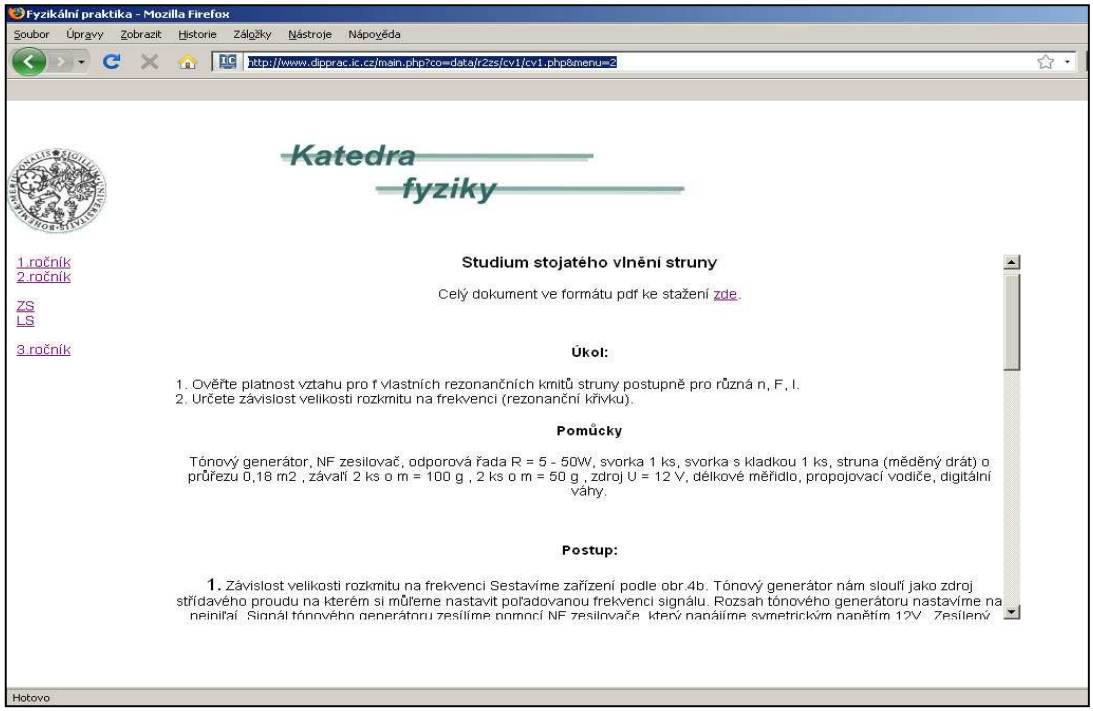
ob. č. 2

4.1.2 Stručný popis datové části

Veškeré materiály k měřením a úlohám jsou uloženy v adresáři ***data***. Ten je dále dělen na podadresáře podle ročníku studia a semestru, ve kterém se jednotlivé předměty vyučují:

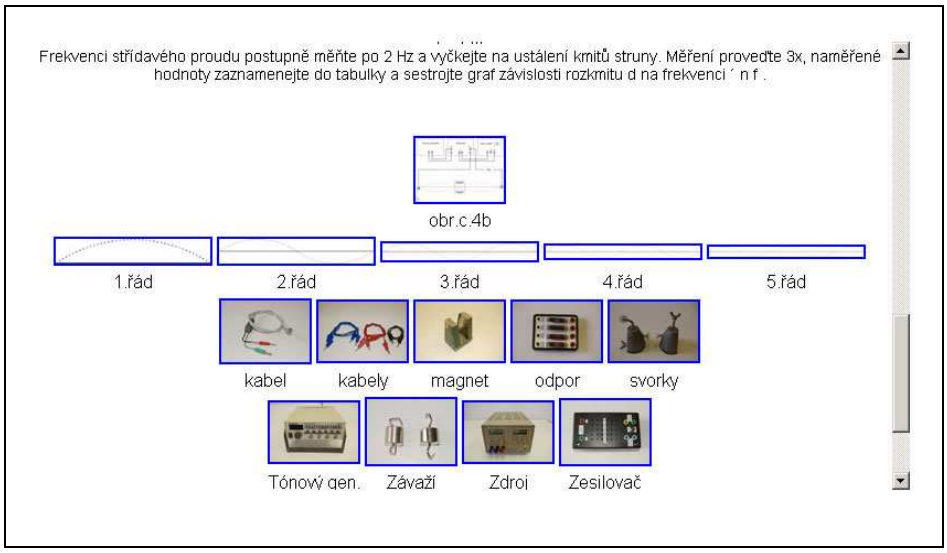
- **r1ls**
- **r1zs**
- **r2ls**
-
-
-

Tyto složky jsou dále členěny na další podadresáře pro jednotlivá měření. V každém je uložen soubor ve formátu pdf s kompletním zněním úlohy, adresář ***obr*** obsahující fotografie, schémata a obrázky k danému měření a soubor ***cvX.php*** zobrazující materiál ve www formě. Písmeno X v názvu souboru značí číslici, ke kterému měření ten daný soubor patří. Tato www prezentace nabízí hned v úvodu možnost stažení celého pdf dokumentu. Dále zobrazuje potřebné informace k vypracování zápisu z měření viz obr. č. 3.



obr. č. 3

V závěru prezentace je u každého měření fotogalerie k danému tématu viz obr. č. 4.



obr. č. 4

Pro náhledy jsou fotky zmenšovány na co nejmenší velikost souboru. Po kliknutí na náhled se otevře nové okno s původní nezmenšenou fotografií.

Část zdrojového kódu pro měření z optiky:

```
<SCRIPT LANGUAGE="JavaScript">
<!--//
function openWindow() {
nahled=window.open("","fotky",'toolbar=0,location=0,scrollbars=0,,resizable=0,top=0,lef
t=0');

with (nahled) { close (onClick); }

}
```

```
//-->
</SCRIPT>

<center>

<h3>Měření s tenkými čočkami</h3>
<p>Celý dokument ve formátu pdf ke stažení <a href="data/r2zs/cv7/cv7.pdf"
target="_blank">zde</a>.</p>
<p> </p>

<h4>Úkol:</h4>
<p align="left">
1) Určete ohniskovou vzdálenost daných čoček.<br>
2) Vypočtenou hodnotu porovnejte s hodnotou určenou graficky.<br>
3) Pokusně ověřte platnost vztahu a´+a 3 4f . (U úkulo 1.)</p>

<h4>Pomůcky</h4>
<p>Optická lavice, sada čoček, měřítko, zdroj světla, clona, stínítko,
předmět.</p>
<p> </p>

<h4>Postup:</h4>
<p>  <font size="+1">1.</font>
Předmět (čtvercový rastr) a stínidlo postavíme na opačné konce optické
lavice a mezi ně postavíme čočku. Světelným zdrojem osvětlíme
předmět a stojánek s čočkou posouváme tak dlouho, až na stínítku
vznikne ostrý obraz předmětu. Na měřidle optické lavice odečítáme
polohu předmětu P, stínítka S a čočky C. Naměřené a vypočtené
hodnoty zapíšeme do tabulky. Měření opakujeme při různých
vzdálenostech předmětu a stínidla (alespoň desetkrát). Ohniskovou
```

vzdálenost pak určíme nejen pomocí vztahu (3), ale i graficky. Hodnoty by měly být vypracovány do tematicky podobné tabulky.

2.
Předmět a stínidlo umístíme na opačné konce optické lavice. Příslušné polohy předmětu a stínidla odečítáme. Čočku posouváme k předmětu tak, až na stínidle vznikne ostrý obraz předmětu. Příslušnou polohu

.
.
.
.

4.2 Přidání dalších měření

Tato část této bakalářské práce je vlastně jen doplněním informací z předešlé kapitoly.

Přidání jednoho měření

Před přidáním jednoho měření je potřeba založit v adresáři pro příslušný semestr složku pro dané měření. Pokud bude v dalších přidávaných měřeních zachován stejný systém ukládání dat, tak se do této složky uloží již dříve zmiňované soubory ve formátu ***pdf***, ***php*** nebo ***html*** a soubor pomocných fotografií do adresáře ***obr.*** Dále je třeba vytvořit odkaz na toto měření do souboru, který se jmenuje stejně jako adresář, v němž jsou uložena data. Tento soubor je umístněn v adresáři ***sites***. Příkladem je již hotový soubor dat

uložených ve složce ***r2zs*** (ročník druhý zimní semestr) a k nim příslušný soubour ***r2zd.php*** v adresáři ***sites***.

Zdrojový kód souboru ***2zs.php:***

```
<center>
<a href="main.php?co=data/r2zs/cv1/cv1.php&menu=2">Studium stojatého
vlnění struny</a><br>
<a href="main.php?co=data/r2zs/cv2/cv2.php&menu=2">Mikroskopická
měření</a><br>
<a href="main.php?co=data/r2zs/cv3/cv3.php&menu=2">Optické laboratorní
metody</a><br>
<a href="main.php?co=data/r2zs/cv4/cv4.php&menu=2">Studium parametrů
dalekohledu</a><br>
```

Podobným způsobem je třeba postupovat při přidávání dat do měření jiných ročníků. Rozdílem je umístění odkazu do souborů pojmenovaných podle toho kterého semestru.

5 ZÁVĚR

Cílem této práce bylo vytvoření souboru dat a informací pomáhajících studentům MVT při cvičení z fyzikálních měření a následném zpracovávání referátu. Jednotlivá měření zpracoval jako součást své bakalářské práce Jan Proll. Z důvodu časové náročnosti při vytváření materiálů je do prezentace vložen pouze soubor dat pro měření jednoho semestru. Chybějící materiály mohou být dodány jako součásti dalších bakalářských prací v následujících ročnících studia.

Při implementaci dat do systému eAmos jsme narazili na jeden malý problém. Příčinou je omezení funkčnosti scriptů php v celém systému eAmos z důvodu bezpečnosti. Proto byla celá prezentace umístěna na jiný server a do eAmosu byl na ní pouze zakomponován odkaz.

V době, kdy jsem volil téma této bakalářské práce, jsem si neuvědomoval rozsah všech prací a hlavně doba, která bude nutná k úpravám všech materiálů. Věřím však, že tento mnou strávený čas přinese dalším studentům ulehčení jejich studií a pomůže při měřeních a vypracovávání referátů.

6 Literatura

Programujeme PHP profesionálně – Jesus Castagneto, Harish Rawat, Sascha Schumann, Chris Scollo, Deepak Veliath

Adobe Photoshop – Martin Vlach

http://www.jaknaweb.cz/

http://www.jakpsatweb.cz/

7 Přílohy

Studium stojatého vlnění struny

Pomůcky: Tónový generátor, NF zesilovač, odporová řada $R = 5 - 50\Omega$, svorka 1 ks, svorka s kladkou 1 ks, struna (měděný drát) o průřezu $0{,}18\ m^2$, závaží 2 ks o $m = 100$ g, 2 ks o $m = 50$ g, zdroj $U = 12$ V, délkové měřidlo, propojovací vodiče, digitální váhy.

Teorie: Kmitočty f_n vlastních rezonančních kmitů struny jsou dány výrazem:

$$f_n = \frac{n}{2 \cdot l} \cdot \sqrt{\frac{F}{\mu}} \qquad 1.$$

Kde n je celé kladné číslo, které udává řád kmitu (tzv. „vyšší harmonické frekvence“), l je délka struny, F je síla napínající strunu
a μ je hmotnost délkové jednotky struny.
Výraz l můžeme snadno odvodit ze známých vztahů:

$$l = n \cdot \frac{\lambda \cdot n}{2} \qquad 2.$$

Kde $\lambda \cdot n$ je vlnová délka příslušná řádu kmitu struny (viz obr.4a)

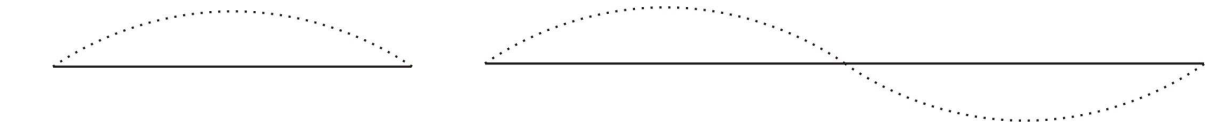
$$f_n = \frac{v}{\lambda \cdot n} \qquad 3.$$

kde v je rychlost šíření vlnění po struně. Rychlost v můžeme rovněž vyjádřit vztahem:

$$v = \sqrt{\frac{F}{\rho \cdot s}} = \sqrt{\frac{F}{\mu}} \qquad 4.$$

Kde F je síla napínající strunu, ρ je hustota struny, s je průřez struny a μ je hmotnost délkové jednotky struny. Z uvedených vztahů 2., 3. a 4. odvodíme vztah 1.

Obr.: 4a

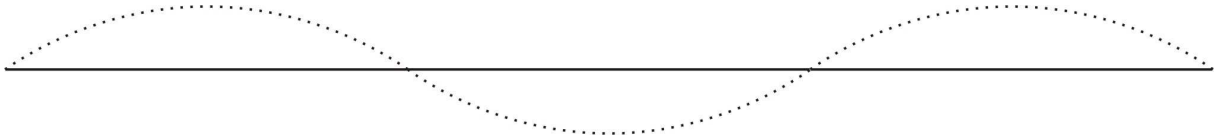


$n = 1$

$\lambda = 2l$

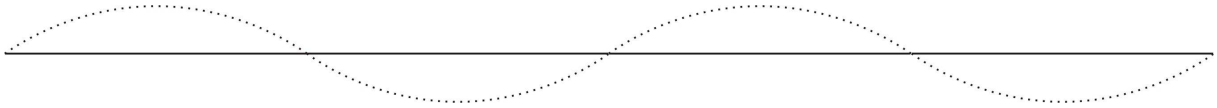
$n = 2$

$\lambda = l$



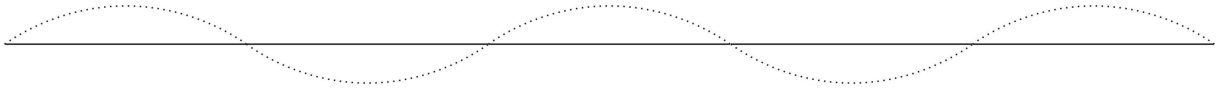
$n = 3$

$\lambda = \frac{2}{3}l$



$n = 4$

$\lambda = \frac{1}{2l}$



$n = 5$

$\lambda = \frac{2}{5}$

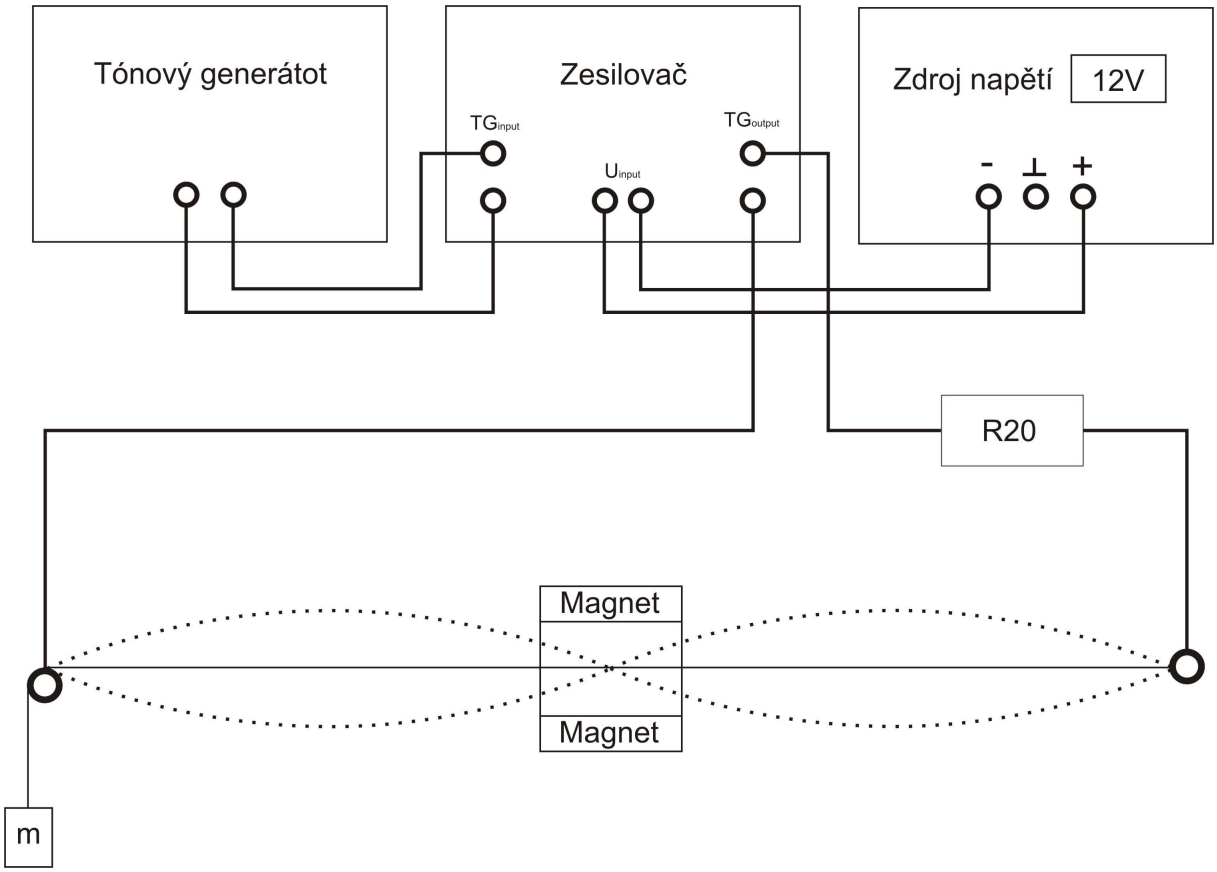
Jestli-že působíme harmonickou silou

$$F_n = F_0 \cdot \sin \cdot 2\pi \cdot f_n' \cdot t$$

na nějaký bod struny mimo uzel, jejíž frekvence odpovídá některé z vlastních frekvencí struny f_n, nastane rezonance a struna začne s touto frekvencí kmitat. Tento výsledek plyne z podrobného rozboru kmitů struny. Z rovnice 1. vyplývá, že při stálé napínací síle F může struna kmitat se základní frekvencí f_1 a s frekvencemi, které jsou celistvým násobkem základní frekvence $f_n = n \cdot f_1$.

Rezonanční kmitočet struny můžeme měnit změnou síly F napínající strunu a změnou délky struny l.

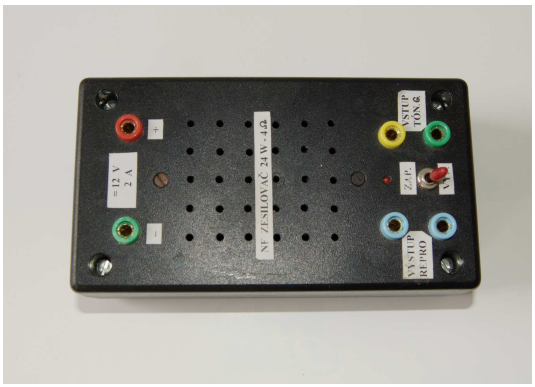
K vyšetřování rezonančních kmitů struny použijeme zařízení znázorněné na obr.4b



Obr.4b

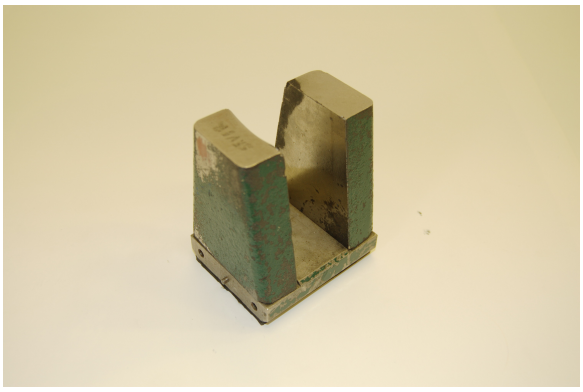
Struna je na jednom konci upevněna a na druhém konci vedena přes kladku a vypínána závažím. Struna je natažena mezi póly permanentního magnetu, s nímž lze podél struny posunovat. Stojaté vlny na struně vzniknou tak, že strunou necháme procházet střídavý proud o známé frekvenci f_n'. V poli magnetu působí na strunu magnetická síla frekvence f_n', mířící kolmo na strunu a kolmo na směr magnetického pole. Rozkmit struny měříme na stupnici umístěné za strunou.

Tónový generátor Goldstar FG-2002C

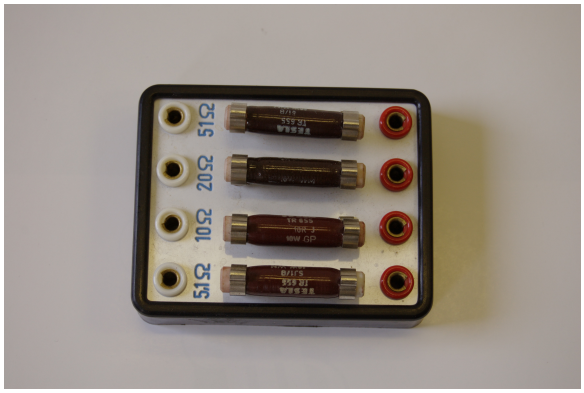


NF zesilovač

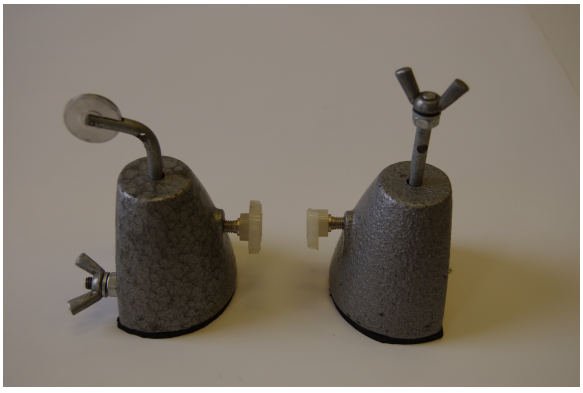
Zdroj napětí BK0181



Permanentní magnet



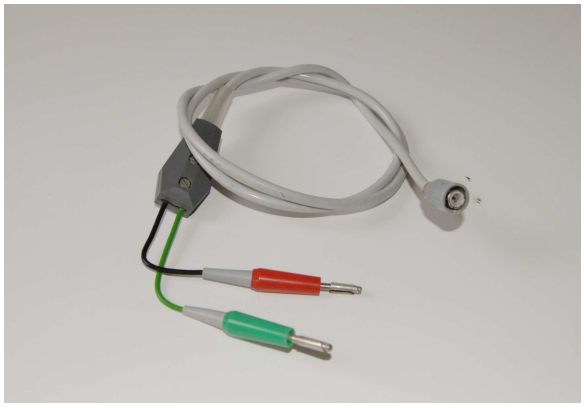
Odporová řada



Svorka s kladkou, svorka

Deska

Závaží



Propojovací kabel s tónovým generátorem

Propojovací kabely

Postup:

1. Závislost velikosti rozkmitu na frekvenci

Sestavíme zařízení podle obr.4b. Tónový generátor nám slouží jako zdroj střídavého proudu na kterém si můžeme nastavit požadovanou frekvenci signálu. Rozsah tónového generátoru nastavíme na nejnižší. Signál tónového generátoru zesílíme pomocí NF zesilovače, který napájíme symetrickým napětím $12V$. Zesílený signál přivedeme přes předřadný odpor na svorky, na kterých je napnuta struna a na jednom konci přes kladku napínána závažím.

Jako výchozí délku struny zvolíme $l = 1\,\mathrm{m}$ a závaží zvolíme o hmotnosti $m = 0{,}1\,\mathrm{kg}$. Postupně hledáme kmity řádu $n = 1, 2,..., 5.$ Pro každé n změříme f_n třikrát a výsledky zpracujeme do tabulky.

č. měření	n	f_n' [Hz] naměřená	f_n [Hz] vypočtená	$\frac{(f_n - f_n')}{f_n} \cdot 100\%$

2. Závislost rezonanční frekvence na napínací síle F

Při měření budeme dodržovat podmínky odpovídající vzniku vždy stejného řádu kmitu.Při ověřování této závislosti umístíme magnet na střed struny a poblíž středu umístíme stupnici. Postupně zatěžujeme strunu různými závažími. Změnou frekvence f_n' střídavého proudu procházejícího strunou dosáhneme rezonance, tj. maximálního rozkmitu struny. Tímto způsobem zjistíme odpovídající si dvojici F a f_n'.

Nastavte tyto parametry: l = 1 m
n = 3
m = 0,050 kg, 0,100 kg, 0,150 kg,
0,200 kg, 0,250 kg

Pro každou sílu proveďte 3 měření a výsledky zpracujte do tabulky.

č. měření	m [kg]	F [N]	f_3' [Hz] naměřená	f_3 [Hz] vypočtená	$\frac{f_3' - f_3}{f_3} \cdot 100\%$

3. Závislost f_n na délce struny l

Při stálé napínací síle F dodržujeme podmínky vzniku stejného řádu. Postupně posouváme upevňovací stojan, magnet a stupnici do odpovídající polohy a měníme tak délku struny l. Změnou frekvence střídavého proudu f_n' pomocí tónového generátoru dosáhneme rezonance. Rezonance je maximální rozkmit.

Nastavte tyto parametry: Řád kmitu n = 2
Závaží m = 0,1 kg
Délka struny l = 0,8 m; 0,9 m; 1,0 m; 1,1 m;
1,2 m

Pro každou délku l proveďte tři měření a výsledky zpracujte do tabulky.

č. měření	$l\,[m]$	$f_2'\,[Hz]$ naměřená	$f_2\,[Hz]$ vypočtená	$\frac{(f_2 - f_2')}{f_2} \cdot 100\%$

4. Závislost velikosti rozkmitu na frekvenci (určení rezonanční křivky)

Maximálního rozkmitu dosáhneme, jestliže strunou prochází proud stejné frekvence jako je frekvence struny určená vztahem l. Za těchto podmínek nastává rezonance. Změřte postupně velikost rozkmitu struny pro frekvence f_n' střídavého proudu od 50 Hz do 75 Hz při těchto parametrech:

n = 2
m = 0,1 kg
l = 1 m

Frekvenci střídavého proudu postupně měňte po 2 Hz a vyčkejte na ustálení kmitů struny. Měření proveďte 3x, naměřené hodnoty zaznamenejte do tabulky a sestrojte graf závislosti rozkmitu d na frekvenci f_n'.

$$d = f(f_n')$$

Úkol:

1. Ověřte platnost vztahu l postupně pro různá n, F, l.
2. Určete závislost velikosti rozkmitu na frekvenci (rezonanční křivku).

Poznámka:

μ určíme jako podíl hmotnosti struny m_s a její délky l

$$\mu = \frac{m_s}{l}$$