
ÚVOD DO SYSTÉMU WEBMATHEMATICA

V tejto kapitole si vysvetlíme základy programovacieho jazyka web*MATHEMATICE* a ukážeme si na jednoduchých ukážkach ako vytvoriť základné typy jsp stránok. Budeme sa venovať jednak možnosti vytvoriť statické stránky, ale aj téme použitia premenných v našich jsp stránkach, zobrazovaniu 2D a 3D grafiky.

Akýkoľvek výpočet, ktorý môžeme uskutočniť v *MATHEMATICE* môžeme uskutočniť aj prostredníctvom stránok web*MATHEMATICE*. Existujú len dve základné obmedzenia. Prvým obmedzením je skutočnosť, že v prípade použitia web*MATHEMATICE* nebudeme mať k dispozícii plnú kapacitu *MATHEMATICA* Front End-u (výpočtového jadra). Druhým obmedzením sú licenčné obmedzenia, ktoré istý typ výpočtov nedovoľujú realizovať.

V tejto sekcii si ukážeme vytvorenie niekoľkých základných typov web stránok. Všetky súbory v tejto kapitole vytvorené, v tvare kompletného projektu pre Workbench si môžete stiahnuť zo servera web*MATHEMATICE* na Strojníckej fakulte STU. V prípade záujmu kontaktujte administrátora servera na adrese mathematica@mathematica.sk.

1 Stránka obsahujúca text a statické výpočty

V tomto príklade vytvoríme niekoľko jednoduchých HTML stránok, ktoré budú obsahovať text a rôzne statické výpočty. Zameriame sa taktiež na možné spôsoby zobrazovania výsledku.

Príklad 1

Vytvorte stránku, ktorá zobrazí výsledok neurčitého integrálu $\int x^2 dx$.

V tomto príklade budeme vytvárať statickú stránku, ktorá po každom spustení zobrazí rovnaký výsledok – výsledok integrovania, funkciu $x^3/3+c$. Výsledok zobrazíme v dvoch obvyklých formátoch – **štandardnom formáte** a v **tradičnom formáte** zápisu.

Nasledujúce tagy slúžia na aktivovanie výpočtového jadra na serveri. Keď servlet container (Apache Tomcat) zistí tieto tagy v zdrojovom kóde stránky, web*MATHEMATICA*, ktorá je na serveri nainštalovaná aktivuje svoje výpočtové jadro a pripraví sa na realizáciu výpočtov.

```
<msp:allocateKernel> a </msp:allocateKernel> .
```

Tagy sú párové, jeden ohraničuje začiatok výpočtového prostredia, druhý ohraničuje jeho koniec (deaktiváciu výpočtového jadra).

Výpočtové prostredie ohraničujeme tagmi

```
<msp:evaluate> a </msp:evaluate> .
```

Medzi tieto ohraničujúce tagy umiestňujeme príkazy jazyka *MATHEMATICA*, ktoré chceme v rámci výpočtov na stránke realizovať. Väčšinou nepotrebujeme zobrazit' všetky výsledky na stránku, preto riadky ukončujeme bodkočiarkou. Riadky ohraničené týmito tagmi sa správajú ako samostatná bunka v *MATHEMATICE*. Môžeme preto do nej umiestniť aj niekoľko príkazov, v správnej štruktúre zodpovedajúcej štruktúre programovacieho jazyka *MATHEMATICE*.

Výsledky výpočtov na stránku zobrazujeme pomocou príkazu **MSPFormat**. Jeho štruktúra je nasledujúca:

```
MSPFormat [premenna, StandardForm] ]
```

alebo

```
MSPFormat [premenna, TraditionalForm] ]
```

Teraz vytvoríme nový projekt v prostredí Workbench (tak, ako sme to popísali v predchádzajúcej kapitole), doplníme riadky zabezpečujúce správne kódovanie diakritiky a potom budeme editovať jednotlivé príkazy.

Pozrime sa na zdrojový dokument tejto jsp stránky:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
  href="/webMathematica/Resources/XSL/mathml.xsl"?>

<%@ page language="java" %>
<%@ page contentType="text/html; charset=utf-8" %>
<%@ taglib uri="/webMathematica-taglib" prefix="msp" %>

<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta name="author" content="Monika Kovacova"/>
    <meta http-equiv="Content-type" content="text/html;
      charset=utf-8" />
    <title>Zobrazenie statického výsledku výpočtu</title>
  </head>

  <body bgcolor="#ffffff">

  <h2>Zobrazenie statického výsledku výpočtu</h2>

    <msp:allocateKernel>

      <msp:evaluate>
        vysledok = Integrate[x^2,x]+c;
      </msp:evaluate>

    <h4>Štandardná forma zobrazenia</h4>
```

```

<msp:evaluate>
    MSPFormat[vysledok, StandardForm]
</msp:evaluate>

<h4>Tradičná forma zobrazenia</h4>
<msp:evaluate>
    MSPFormat[vysledok, TraditionalForm]
</msp:evaluate>

</msp:allocateKernel>

</body>

</html>

```

Projekt, vrátane všetkých zdrojových dokumentov, je dostupný na stiahnutie na webovom serveri s označením **priklad01** .

Podstatné časti tohto zdrojového dokumentu sú vyznačené tučným typom písma. Poďme teraz tento dokument analyzovať. Začiatok a koniec výpočtovej časti sú ohraničené tagmi `<msp:allocateKernel>` a `</msp:allocateKernel>`. V prvom výpočtovom prostredí sme definovali premennú `vysledok` a priradili jej hodnotu výsledku výpočtu integrálu `Integrate[x^2,x]+c`. Pri každom zobrazení tejto stránky sa premennej `vysledok` priradí rovnaká hodnota.

Informáciu o spôsobe zobrazenia výsledku výpočtu sme umiestnili do tagov `<h4>` a `</h4>`. Ide o HTML tagy, ktoré ohraničujú nadpis štvrtej úrovne.

V nasledujúcom výpočtovom prostredí sme použili príkaz pre zobrazenie výsledku a jeho sformátovanie v štandardnom formáte.

```

<msp:evaluate>
    MSPFormat[vysledok, StandardForm]
</msp:evaluate>

```

Postup sme zopakovali a výsledok sme sformátovali v tradičnom formáte.

```

<msp:evaluate>
    MSPFormat[vysledok, TraditionalForm]
</msp:evaluate>

```

Výsledná stránka vyzerá takto:



Všimnite si, že výsledok v prípade štandardného formátu je okrem zobrazenia ešte aj usporiadaný podľa svojej lexikálnej štruktúry. V prípade tradičnej formy je výsledok zobrazený tak, ako sme ho definovali. Tradičná forma je pre čitateľa prehľadnejšia.

Oba spôsoby zobrazenia výsledku zobrazujú matematické vzťahy v tvare obrázkov, t.j. server vygeneruje obrázky a pošle ich späť prehliadaču užívateľa, ktorý ich zobrazí. Táto forma zobrazenia výsledku je úplne bezproblémová z hľadiska užívateľa a prehliadača. Akýkoľvek starý prehliadač zobrazí obrázok správne. Problém s touto formou zobrazovania výsledkov pocítíme v tom okamihu, keď sa pokúsime do prehliadača zaslať napríklad výsledok veľmi zložitého integrálu – veľmi dlhý obrázok. Prehliadač zobrazí posuvníky, ktoré obvykle spôsobia úplné znehodnotenie grafického dizajnu stránky.

Jedným z možných riešení je zobrazovať výsledky pomocou MathML kódovania. Ide o spôsob zápisu matematických vzťahov pomocou špeciálnych tagov. Podobnejšie sa tejto otázke venujeme v samostatnej kapitole. V nasledujúcom príklade ukážeme len základné možnosti webMATHMATICE pri zobrazovaní výsledku výpočtu pomocou tohto kódovania.

Príklad 2

Vytvorte stránku, ktorá zobrazí výsledok roznásobenia $\left(x + \frac{x}{y^2}\right)^3$ pomocou MathML kódu.

V tomto príklade budeme opäť vytvárať statickú stránku, ktorá po každom spustení zobrazí rovnaký výsledok.

Výsledok výpočtu, kódovaný pomocou MathML, môžeme zobraziť pomocou príkazu **MSPFormat**. Jeho štruktúra je nasledujúca:

```
MSPFormat [premenna, TraditionalForm, PresentationMathML]
```

alebo

```
MSPFormat [premenna, TraditionalForm, ContentMathML]
```

alebo

```
MSPFormat [premenna, MathMLForm]
```

Teraz vytvoríme nový projekt v prostredí Workbench, doplníme riadky zabezpečujúce správne kódovanie diakritiky a potom budeme editovať jednotlivé príkazy.

Pozrime sa na zdrojový dokument tejto jsp stránky:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
  href="/webMathematica/Resources/XSL/mathml.xsl"?>

<%@ page language="java" %>
<%@ page contentType="text/html; charset=utf-8" %>
<%@ taglib uri="/webMathematica-taglib" prefix="msp" %>

<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta name="author" content="Monika Kovacova"/>
    <meta name="copyright" content="Monika Kovacova,
      mathematica@mathematica.sk"/>
    <meta http-equiv="Content-type" content="text/html;
      charset=utf-8" />
    <title>Zobrazenie statického výsledku výpočtu</title>
```

```

</head>

<body bgcolor="#ffffff">
<h2>Zobrazenie statického výsledku výpočtu</h2>

<msp:allocateKernel>

  <msp:evaluate>
    MSPPageOptions[ "ContentType" -> "text/xml" ];
  </msp:evaluate>

  <msp:evaluate>
    vysledok = Expand[(x+x/y^2)^3];
  </msp:evaluate>

  <h4>Tradičná forma zobrazenia</h4>
  <msp:evaluate>
    MSPFormat[vysledok,TraditionalForm]
  </msp:evaluate>

  <h4>Tradičná forma zobrazenia - PresentationMathML</h4>
  <msp:evaluate>
    MSPFormat[vysledok,TraditionalForm,PresentationMathML]
  </msp:evaluate>

  <h4>Tradičná forma zobrazenia - ContentMathML</h4>
  <msp:evaluate>
    MSPFormat[vysledok,TraditionalForm,ContentMathML]
  </msp:evaluate>

  <h4>Čistá MathML forma zobrazenia</h4>
  <msp:evaluate>
    MSPFormat[vysledok,MathMLForm]
  </msp:evaluate>

</msp:allocateKernel>

</body>

</html>

```

Projekt, vrátane všetkých zdrojových dokumentov, je dostupný na stiahnutie na webovom serveri s označením **príklad02**.

Podstatné časti tohto zdrojového dokumentu sú vyznačené tučným typom písma. Poďme teraz tento dokument analyzovať. Na rozdiel od predchádzajúceho príkladu potrebujeme definovať charakter obsahu vytvárajúcej stránky. Služi na to príkaz

```
MSPPageOptions[ "ContentType" -> "text/xml" ];
```

Tento príkaz je potrebné umiestniť do výpočtového prostredia webMathematica, pretože sa uplatňuje len vtedy, keď je stránka odoslaná na výpočet.

Druhou nutnou súčasťou, aby zobrazovanie stránky bolo korektné je definovanie štýlu v úvode stránky. Tento riadok sme pridávali aj v predchádzajúcich dokumentoch – bola to predpríprava pre správne zobrazovanie výsledku v prehliadači užívateľa. Aj keď je možné použiť aj iné súbory pre popis MathML kódovania v rámci stránky, odporúčame používať súbor **mathml.xsl** umiestnený priamo v štruktúre webMATHEMATICE

```
<?xml-stylesheet type="text/xsl"
  href="/webMathematica/Resources/XSL/mathml.xsl"?>
```

WebMATHMATICA podporuje všetky tri spôsoby MathML kódovania výrazov.

V našom príklade sme použili tieto príkazy pre zobrazenie výsledku a jeho sformátovanie.

```
<<msp:evaluate>
  MSPFormat[vysledok, TraditionalForm, PresentationMathML]
</msp:evaluate>
```

```
<msp:evaluate>
  MSPFormat[vysledok, TraditionalForm, ContentMathML]
</msp:evaluate>
```

```
<msp:evaluate>
  MSPFormat[vysledok, MathMLForm]
</msp:evaluate>
```

Výsledná stránka vyzerá takto:



Všimnite si, že zobrazenie výsledku výpočtu pomocou kontextového spôsobu kódovania MathML rešpektuje lexikálnu stránku výrazu a preto je výsledok usporiadaný vzhľadom na jeho lexikálnu štruktúru.

Na rozdiel od obrázkov môže prísť, najmä u starších prehliadačov, ku problémom so správnym zobrazením matematických výrazov. V čase písania tejto knihy už bol plug-in **MathPlayer** zaradený medzi automatické updaty systému Windows a tak je reálny predpoklad, že každý užívateľ, ktorý používa ako prehliadač stránok niektorú verziu Internet Exploreru, ho už má nainštalovaný. Prehliadače Mozilla vyšších verzií žiadne plug-iny pre korektné zobrazenie MathML kódu nepotrebujú. Na mnohých počítačoch je však problém s fontmi a pri požiadavke na zobrazenie takejto stránky je užívateľ požiadaný o doinštalovanie fontov.

Aj napriek problémom, ktoré môžu vzniknúť pri zobrazení stránok v prehliadači prináša tento spôsob zobrazovania matematiky tak veľké množstvo výhod, že ho môžeme len odporučiť. Ak váš prehliadač (váš WorkBench) nezobrazil správne predchádzajúcu ukážku, nainštalujte si do systému plug-in MathPlayer. (Predpokladáme, že ako hlavný prehliadač používate Internet Explorer a máte ho nastavený aj vo vlastnostiach projektu, v ktorom práve pracujete.) Tento plug-in si môže stiahnuť napríklad z adresy

<http://www.dessci.com/en/products/mathplayer/download.htm>

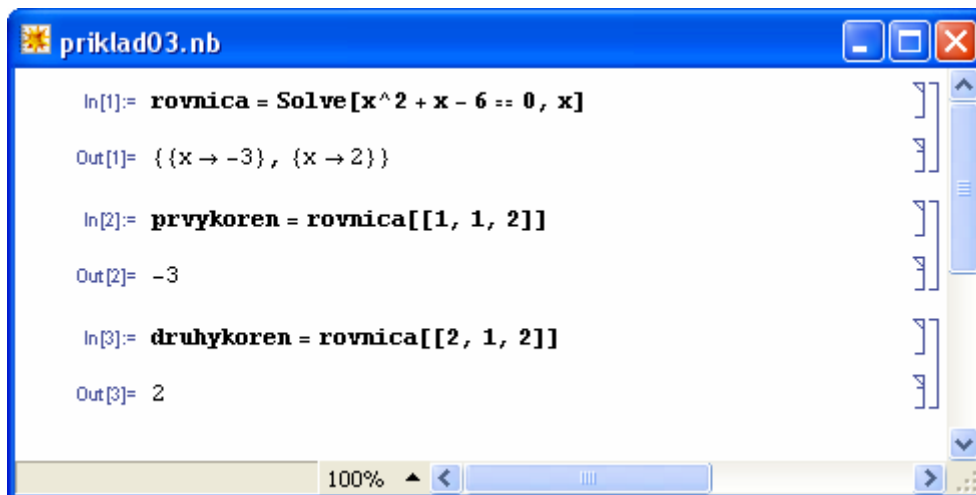
Inštalácia je úplne bezproblémová. Pred inštaláciou zatvorte Workbench aj všetky otvorené stránky vo vašom prehliadači. Po dokončení inštalácie pokračujte normálne v práci na projekte.

Príklad 3

Vytvorte statickú stránku, ktorá obsahuje zložitejší výpočet programovaný v jazyku *MATHEMATICA* a výsledok následne zobrazí prostredníctvom web stránky.

Vhodným zadaním pre tento typ príkladu je ukážka, v ktorej vypočítame korene konkrétnej kvadratickej rovnice pomocou príkazu `Solve` a následne tieto korene zo získaného výsledku separujeme a samostatne umiestnime na web stránku.

Ak by sme úlohu riešili priamo v prostredí *MATHEMATICA*, použili by sme nasledujúce príkazy.



```
in[1]:= rovnica = Solve[x^2 + x - 6 == 0, x]
Out[1]= {{x -> -3}, {x -> 2}}

in[2]:= prvkykoren = rovnica[[1, 1, 2]]
Out[2]= -3

in[3]:= druhykoren = rovnica[[2, 1, 2]]
Out[3]= 2
```

V tomto príklade budeme opäť vytvárať statickú stránku, ktorá po každom spustení zobrazí rovnaký výsledok. **Cieľom príkladu je vysvetliť, ako budeme v prostredí webovej stránky pracovať s viacerými „MATHEMATICA“ premennými.**

Pozrime sa na zdrojový dokument tejto jsp stránky:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
  href="/webMathematica/Resources/XSL/mathml.xsl"?>
```

```

<%@ page language="java" %>
<%@ page contentType="text/html;charset=utf-8" %>
<%@ taglib uri="/webMathematica-taglib" prefix="msp" %>

<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta name="author" content="Monika Kovacova"/>
    <meta http-equiv="Content-type" content="text/html;
      charset=utf-8" />
    <title> Použitie viacerých premenných vo výpočte </title>
  </head>

  <body bgcolor="#ffffff">
    <h2>Použitie viacerých premenných vo výpočte</h2>

    <msp:allocateKernel>
      <msp:evaluate>
        MSPPageOptions[ "ContentType" -> "text/xml"];
      </msp:evaluate>

      <msp:evaluate>
        rovnica = Solve[x^2+x-6 == 0,x];
      </msp:evaluate>

      <msp:evaluate>
        prvykoren = rovnica[[1,1,2]];
        druhykoren = rovnica[[2,1,2]];
      </msp:evaluate>

    <p>Prvým koreňom rovnice
      <math xmlns='http://www.w3.org/1998/Math/MathML'
        mathematica:form='StandardForm'
        xmlns:mathematica='http://www.wolfram.com/XML/'>
        <mrow>
          <mrow>
            <msup>
              <mi>x</mi>
              <mn>2</mn>
            </msup>
            <mo>+</mo>
            <mi>x</mi>
            <mo>-</mo>
            <mn>6</mn>
          </mrow>
          <mo>&#10869;</mo>
          <mn>0</mn>
        </mrow>
      </math> je číslo
      <msp:evaluate>
        MSPFormat[prvykoren,MathMLForm]
      </msp:evaluate>. Druhým koreňom rovnice je číslo
      <msp:evaluate>
        MSPFormat[druhykoren,MathMLForm]
      </msp:evaluate>.
    </p>

  </msp:allocateKernel>

</body>
</html>

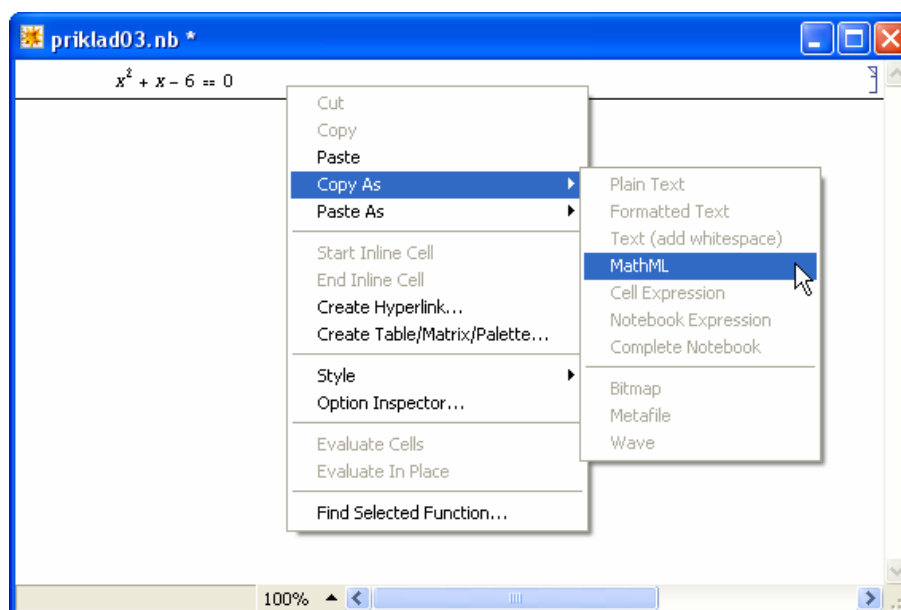
```


Projekt, vrátane všetkých zdrojových dokumentov, je dostupný na stiahnutie na webovom serveri s označením **priklad03**.

Podstatné časti tohto zdrojového dokumentu sú vyznačené tučným typom písma. Poďme teraz tento dokument analyzovať. **Cieľom tohto príkladu bolo ukázať, ako je možné v prostredí webMATHEMATICE pracovať s viacerými premennými.**

V tomto príklade pracujeme s *MATHEMATICA* premennými **rovnica**, **prvykoren** a **druhykoren**. Všetky tri premenné by sme mohli umiestniť aj do jedného prostredia na výpočet ohraničeného tagmi `<msp:evaluate>` a `</msp:evaluate>`, ale v ukážke sme zámerne zvolili menej efektívny spôsob – umiestnenia do dvoch samostatných prostredí ohraničených týmito tagmi. Demonštrovali sme tak skutočnosť, že **premenné, ktoré sú raz vytvorené v rámci webovej stránky majú globálnu platnosť v rámci tejto webovej stránky** a to bez ohľadu na to, v ktorom výpočtovom prostredí ich použijeme a koľko krát ich hodnotu zavoláme.

Ďalšou zaujímavosťou tejto ukážky je zapísanie časti stránky pomocou MathML kódu. Konkrétne sme pomocou tohto spôsobu kódovania zapísali kvadratickú rovnicu $x^2 + x - 6 = 0$. Nie je potrebné učiť sa podrobne tento kódovací jazyk. Môžeme využiť programový systém *MATHEMATICA*. Otvoríme opäť notebook `priklad03.nb` a zapíšeme do neho požadovanú kvadratickú formulu. Potom označíme príslušnú bunku a po kliknutí na pravé tlačidlo myši s kontextovej ponuky vyberieme **Copy as >> MathML**



Skopírovaný obsah položíme (Ctrl + v) do zdrojového dokumentu `priklad03.jsp`.

Celý text, ktorý chceme na tejto stránke zobraziť sme ohraničili tagmi `<p>` a `</p>`. Tento spôsob ohraničenia textu zabezpečí, že celý text ohraničený týmto spôsobom sa zobrazí na webovej stránke v rámci jedného paragrafu. Všimnite si, že nie je podstatné, kde začína text a kde výpočtové prostredie. V rámci jedného riadku obe prostredia môžeme kombinovať podľa našich potrieb.

Všimnite si, že premenné **prvykoren** a **druhykoren**, ktoré sme definovali v samostatnom výpočtovom prostredí ohraničenom tagmi `<msp:evaluate>` a `</msp:evaluate>` na začiatku stránky, môžeme použiť v inom výpočtovom prostredí ako výstup výpočtu.

Výsledná stránka vyzerá nasledovne.



Zmenou rovnice v zdrojovom dokumente stránky bez akýchkoľvek ďalších zmien dostaneme riešenie inej kvadratickej rovnice. Neskôr si ukážeme, ako je možné aj priamo v rámci web stránky generovať MathML kód z nejakej *MATHEMATICA* premennej. Potom už nebude nutné používať **Copy as >> MathML** v prostredí *MATHEMATICA*. Napríklad:

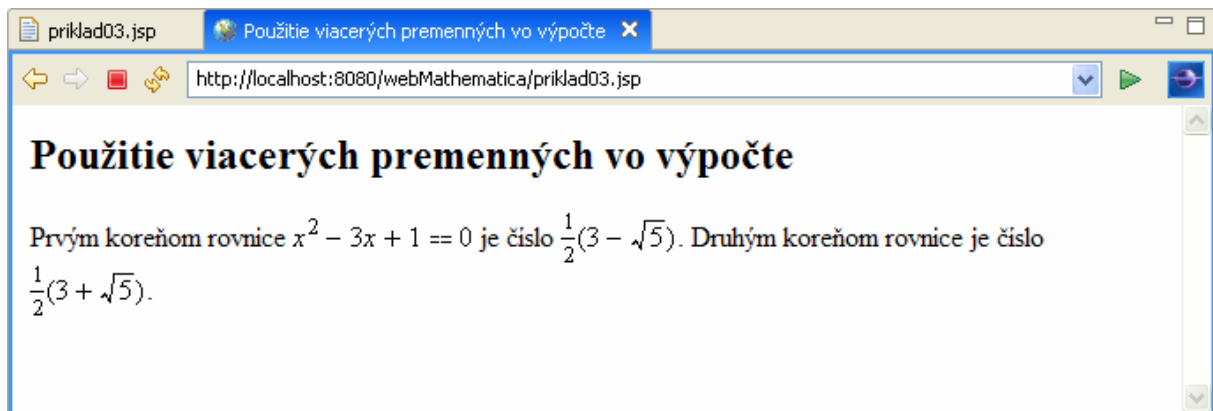
```
<msp:evaluate>
    rovnica=Solve[x^2-3x+1 == 0,x];
</msp:evaluate>

<msp:evaluate>
    prvykoren=rovnica[[1,1,2]];
    druhykoren=rovnica[[2,1,2]];
</msp:evaluate>

<p>Prvým koreňom rovnice
    <math xmlns='http://www.w3.org/1998/Math/MathML'
        mathematica:form='StandardForm'
        xmlns:mathematica='http://www.wolfram.com/XML/'>
        <mrow>
            <mrow>
                <msup>
                    <mi>x</mi>
                    <mn>2</mn>
                </msup>
                <mo>-</mo>
                <mrow>
                    <mn>3</mn>
                    <mo>&#8290;</mo>
                    <mi>x</mi>
                </mrow>
                <mo>+</mo>
                <mn>1</mn>
            </mrow>
            <mo>&#10869;</mo>
            <mn>0</mn>
        </mrow>
    </math>
    je číslo
    <msp:evaluate>
        MSPFormat[prvykoren,MathMLForm]
    </msp:evaluate>. Druhým koreňom rovnice je číslo
    <msp:evaluate>
        MSPFormat[druhykoren,MathMLForm]
    </msp:evaluate>.
```

</p>

Výsledná stránka vyzerá v tomto prípade nasledovne.



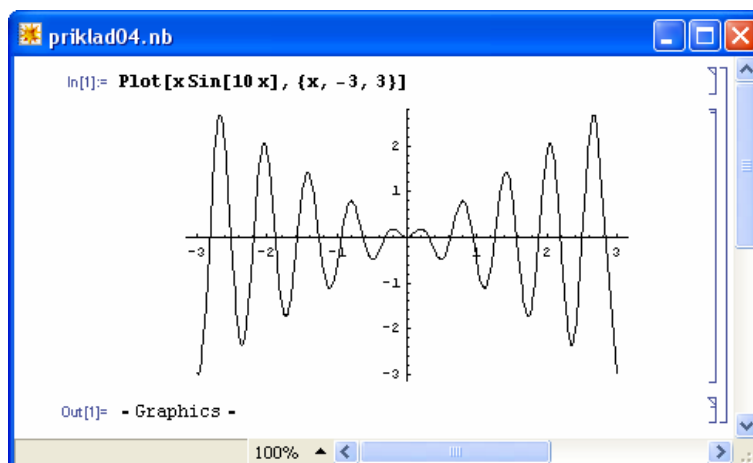
2 Stránka obsahujúca grafiku a statické výpočty

V tejto sekcii vytvoríme niekoľko jednoduchých HTML stránok, ktoré budú obsahovať grafickú prezentáciu vypočítaného výsledku – statického výpočtu. Ukážeme si príklady, pomocou ktorých je možné prezentovať v prostredí webovej stránky 2D graf a 3D grafy.

Príklad 4

Vytvorte statickú stránku, ktorá zobrazí prostredníctvom web stránky 2D graf, ako výsledok príkazu `Plot`.

V tomto príklade zobrazíme graf funkcie $f : y = x \cdot \sin(10x)$ na intervale $[-3, 3]$. Ak by sme úlohu riešili priamo v prostredí systému *MATHEMATICA*, použijeme príkaz `Plot`.



V tomto príklade budeme teda opäť vytvárať statickú stránku, ktorá po každom spustení zobrazí rovnaký výsledok. **Cieľom príkladu je vysvetliť, ako môžeme v prostredí webovej stránky zobrazit' 2D obrázok ako výsledok výpočtu.**

Pozrime sa na zdrojový dokument tejto jsp stránky:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
href="/webMathematica/Resources/XSL/mathml.xsl"?>

<%@ page language="java" %>
<%@ page contentType="text/html; charset=utf-8" %>
<%@ taglib uri="/webMathematica-taglib" prefix="msp" %>

<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta name="author" content="Monika Kovacova"/>
    <meta http-equiv="Content-type" content="text/html;
charset=utf-8" />
    <title>Zobrazenie 2D grafu funkcie</title>
  </head>

  <body bgcolor="#ffffff">
    <h2>Zobrazenie 2D grafu funkcie</h2>

    <msp:allocateKernel>

      <msp:evaluate>
        graf = Plot[x Sin[10x], {x, -3, 3}];
      </msp:evaluate>

      <p> Grafom funkcie
      <math xmlns='http://www.w3.org/1998/Math/MathML'
        mathematica:form='TraditionalForm'
        xmlns:mathematica='http://www.wolfram.com/XML/'>
        <mtext>f : y = x sin(10 x)</mtext>
      </math> je
      </p>
      <msp:evaluate>
        MSPShow[graf]
      </msp:evaluate>

    </msp:allocateKernel>

  </body>
</html>
```

Projekt, vrátane všetkých zdrojových dokumentov, je dostupný na stiahnutie na webovom serveri s označením **príklad04** .

Rovnako ako v predchádzajúcich príkladoch sme tučným fontom označili dôležité časti tohto zdrojového dokumentu. V prvom výpočtovom prostredí, medzi tagmi **<msp:evaluate>a </msp:evaluate>**sme definovali premennú **graf** a priradili sme jej obrázok – ako výsledok príkazu **Plot**.

V nasledujúcej časti sme do HTML stránky uviedli ukážku zápisu matematického textu $f : y = x \cdot \sin(10x)$ v tradičnom tvare zápisu (Traditional Form). Použili sme rovnako, ako v predchádzajúcom príklade postup **Copy as >> MathML**. Bunka, z ktorej sme text kopirovali

bola sformátovaná do tvaru tradičnej formy zápisu priamo v prostredí *MATHEMATICE* (súbor `priklad04.nb`).

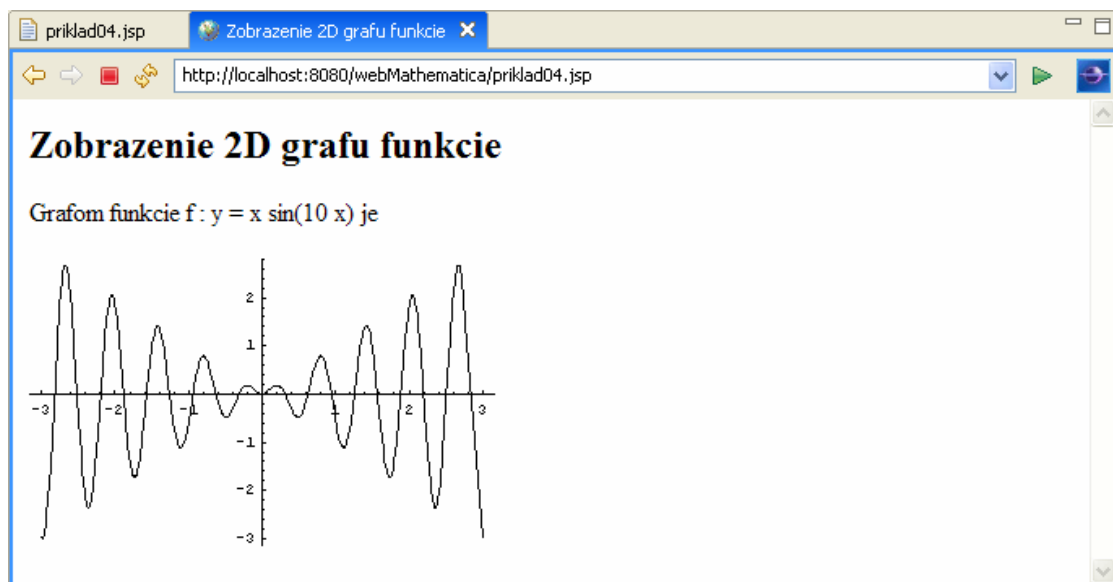
Novým príkazom, ktorý sa objavil v ďalšej časti zdrojového kódu bol príkaz

MSPShow[graficky vystup]

Tento príkaz, ako parameter požaduje výstup v tvare `Graphics`.

Situácia so zobrazením obrázkov na webovej stránke nie je tak jednoduchá, ako je to v prípade zobrazenia textu. Výpočtové jadro *MATHEMATICE* na serveri zrealizuje výpočet, t.j. vytvorí objekt typu `Graphics` (vytvorí obrázok, ktorý je potrebné zaslať späť užívateľovi a následne ho zobrazí v jeho webovom prehliadači). Pretože HTML je bezstavový protokol, ktorý je schopný zasielať len text medzi serverom a webovým prehliadačom užívateľa, nie je možné priamo zaslať do prehliadača užívateľa obrázok. Vytvorený obrázok sa preto uloží vo formáte `gif` ako dočasný súbor na serveri a užívateľovi je zaslaný tag `` s adresou tohto dočasného súboru. Jeho prehliadač následne tento obrázok zobrazí.

Celý tento postup sa deje bez priameho zásahu užívateľa a o jeho správnu realizáciu sa stará príkaz `MSPShow`. Obrázky preto nemôžeme zobrazovať pomocou príkazu `MSPFormat`. Výsledok našej požiadavky vidíte na nasledujúcom obrázku.



Pomocou príkazu `MSPShow` môžeme na webovú stránku zaslať akýkoľvek objekt typu `Graphics`. Tento príkaz môžeme použiť aj pri zobrazení výsledku pomocou príkazu `ListPlot`, `ParametricPlot` a ďalších.

Grafický výsledok (farba, pozadie, hrúbka čiary, veľkosť obrázku...) môžeme ovplyvňovať buď ešte pred jeho vygenerovaním, pomocou príkazov *MATHEMATICE*, alebo neskôr, počas jeho zobrazovania na webovej stránke. Ak chceme dosiahnuť čo najlepší výsledok (z hľadiska kvality výstupného obrázku) je vhodné parametre obrázku nastaviť ešte pred jeho vygenerovaním.

Na našom príklade si ukážeme jednoduchý spôsob, ako ovplyvniť kvalitu výstupu. Do zdrojového dokumentu príkladu doplníme niektoré parametre, napríklad:

```
<msp:evaluate>
  graf = Plot[x Sin[10x], {x, -3, 3},
```

```

PlotStyle -> {Thickness[0.01], Dashing[{0.02}], Hue[0.6]},
PlotPoints -> 80, Background -> GrayLevel[0.9],
PlotLabel -> "Graf funkcie"];

```

</msp:evaluate>

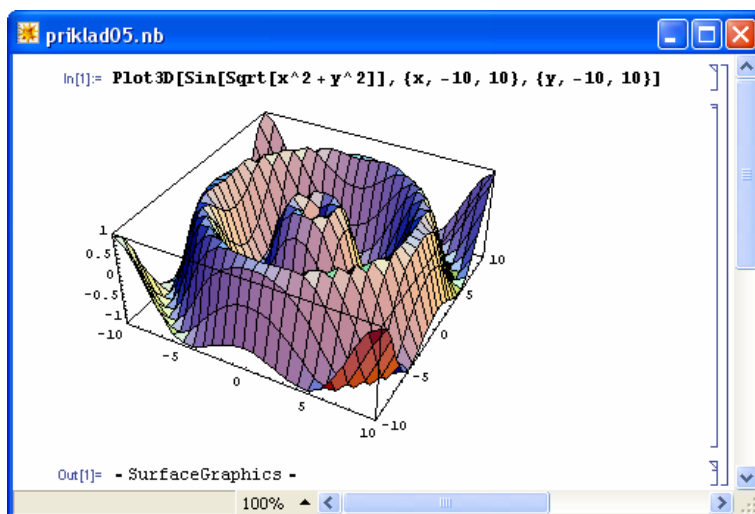
Výsledkom je potom takýto výsledok.



Príklad 5

Vytvorte statickú stránku, ktorá zobrazí prostredníctvom web stránky 3D graf, ako výsledok príkazu `Plot3D`.

V tomto príklade zobrazíme graf funkcie $f: z = \sin(\sqrt{x^2 + y^2})$ na intervale $[-10, 10] \times [-10, 10]$. Ak by sme úlohu riešili priamo v prostredí systému *MATHEMATICA*, použijeme príkaz `Plot3D`.



V tomto príklade budeme teda opäť vytvárať statickú stránku, ktorá po každom spustení zobrazí rovnaký výsledok. **Cieľom príkladu je vysvetliť, ako môžeme v prostredí webovej stránky zobrazit' 3D obrázok ako výsledok výpočtu. Situácia je trochu zložitejšia, ako v prípade 2D objektu.** Musíme preto postupne analyzovať možné príčiny a ich uviesť ich riešenia.

Pozrime sa na zdrojový dokument tejto jsp stránky:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
href="/webMathematica/Resources/XSL/mathml.xsl"?>

<%@ page language="java" %>
<%@ page contentType="text/html; charset=utf-8" %>
<%@ taglib uri="/webMathematica-taglib" prefix="msp" %>

<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta name="author" content="Monika Kovacova"/>
    <meta http-equiv="Content-type" content="text/html;
charset=utf-8" />
    <title>Zobrazenie 3D grafu funkcie</title>
  </head>

  <body bgcolor="#ffffff">
<h2>Zobrazenie 3D grafu funkcie</h2>

<msp:allocateKernel>

<msp:evaluate>
  graf = Plot3D[Sin[Sqrt[x^2 + y^2]], {x, -10, 10}, {y, -10, 10}];
</msp:evaluate>

<p> Grafom funkcie je </p>
<msp:evaluate>
  MSPLive3D[graf]
</msp:evaluate>

</msp:allocateKernel>

</body>

</html>
```

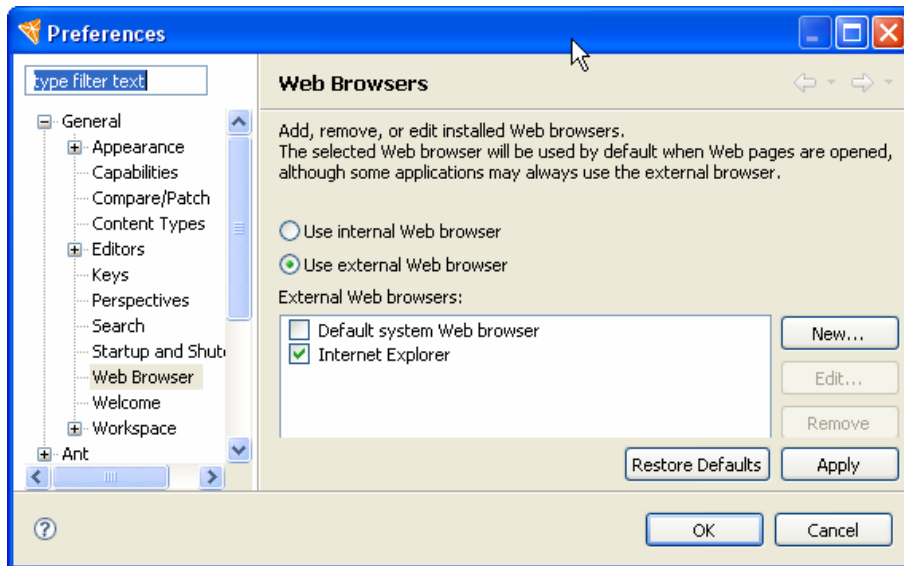
Projekt, vrátane všetkých zdrojových dokumentov, je dostupný na stiahnutie na webovom serveri s označením **príklad05**.

Rovnako ako v predchádzajúcich príkladoch sú dôležité časti zdrojového kódu vyznačené tučne. V prvom výpočtovom prostredí sme definovali grafický objekt typu **-Surface Graphics-**. V druhom výpočtovom prostredí sme graf zobrazili pomocou príkazu

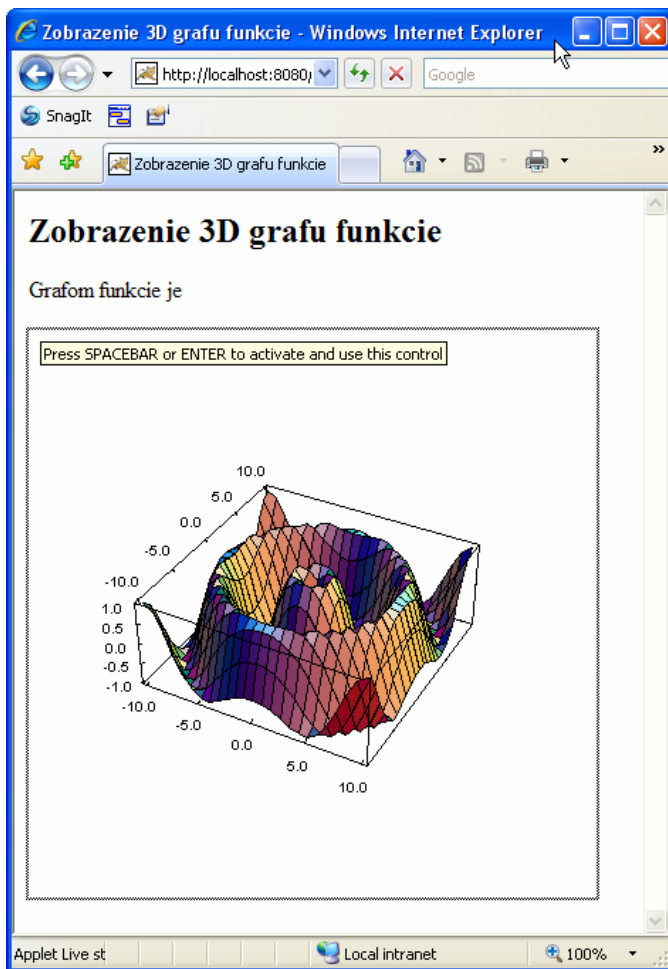
MSPLive3D[objekt 3D]

Objektom typu 3D môže byť buď *MATHEMATICA* objekt typu **-Surface Graphics-**, alebo objekt typu **-Graphics3D-**. Rovnako ako v prípade 2D je výsledkom objekt, ktorý je uložený v dočasnom adresári na serveri a do prehliadača užívateľa je zasielaný len tag umožňujúci zobrazit' Live Graphics 3D applet.

Tento postup však nie je celkom korektný, ak ho chceme použiť v rámci pracovnej plochy Workbenchu. Pri pokuse o preklad a zobrazenie tohto súboru príde ku pádu celého Workbenchu. Táto chyba programu zatiaľ ešte nebola odstránená a preto skôr, ako budeme chcieť vytvorený dokument zobrazit', **musíme spôsob zobrazenia prepnúť na zobrazovanie v externom okne prehliadača**. Nastavenia zmeníme **Window >> Preferences >> General >> Web Browser** takto



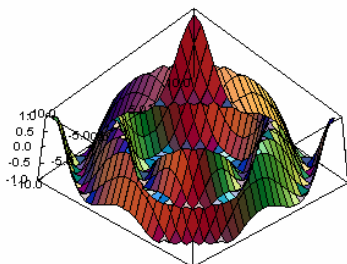
Po zmene pripravený dokument priklad05.jsp uploadneme na lokálny server a overíme jeho funkčnosť.



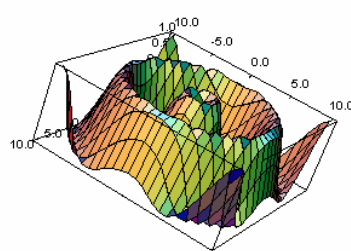
Vytvorená web stránka sa nám zobrazí v samostatnom okne prehliadača, mimo prostredia Workbenchu a ak máme povolené použitie Java appletov na webových stránkach, na vytvorenom obrázku sa zobrazí text s výzvou na aktiváciu tohto appletu.

Po aktivácii môžeme využívať rôzne interaktívne vlastnosti tohto appletu, napríklad pohybom myši môžeme obrázok otočiť tak, aby sme ho videli z požadovaného uhla pohľadu, prudkým pohybom myši v niektorom zo smerov môžeme zaktivovať animáciu, súčasným stlačením klávesy CTRL a pohybom myši mení spôsob zobrazenia a podobne.

otočenie myšou

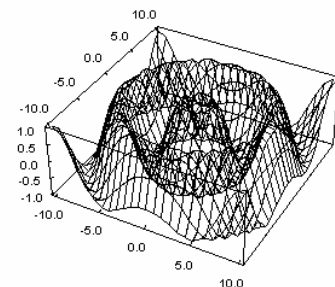


pohyb objektu v smere osi \vec{x} - stlačenie klávesy X (v jednom smere), Shift + X v opačnom smere

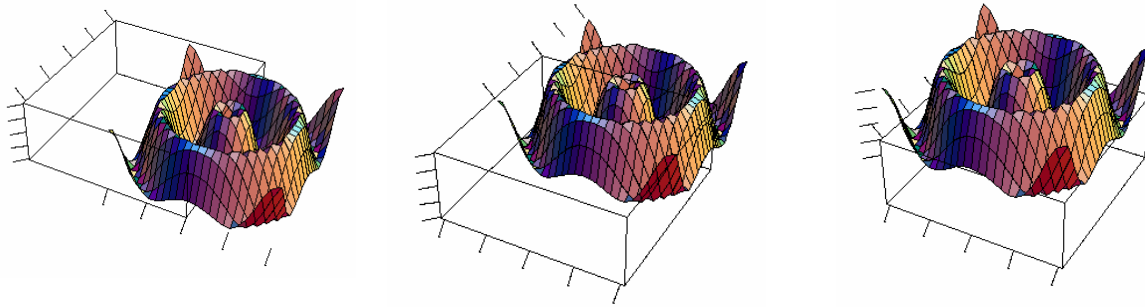


pohyb objektu v smere osi \vec{y} - stlačenie klávesy Y (v jednom smere), Shift + Y v opačnom smere

stlačenie klávesy F



pohyb objektu v smere osi \vec{z} - stlačenie klávesy Z (v jednom smere), Shift + Z v opačnom smere



Ak finálnu vytvorenú stránku uploadneme na server, stránka bude zobrazená korektne. problém s nastavením, ktorý sme popísali v predchádzajúcej časti sa týka len práce v prostredí Workbench. Je veľmi pravdepodobné, že v niektorej z nasledujúcich verzií bude tento problém odstránený.

Ďalšie problémy pri zobrazovaní 3D objektov pomocou tohto appletu nastanú, ak budeme chcieť v obsahu stránky použiť aj text, obsahujúci MathML kód. Táto kombinácia nebude fungovať. Doplňme predchádzajúcu ukážku takto:

```

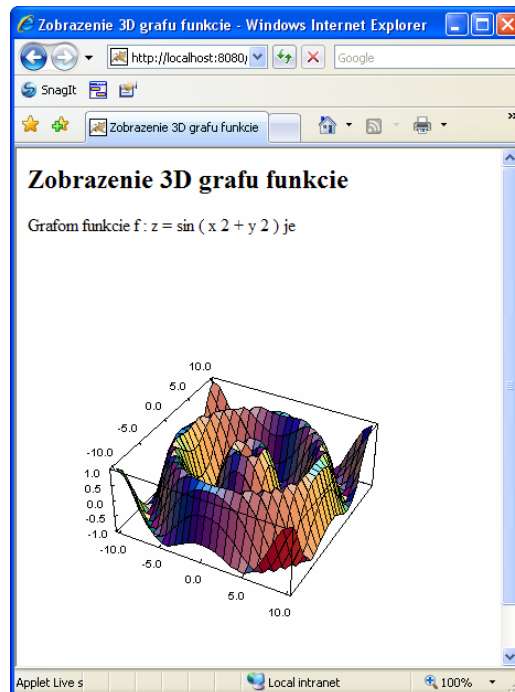
...
<msp:evaluate>
  graf = Plot3D[Sin[Sqrt[x^2 + y^2]], {x, -10, 10}, {y, -10, 10}];
</msp:evaluate>

<p> Grafom funkcie
  <math xmlns='http://www.w3.org/1998/Math/MathML'>
    <mrow>
      <mrow>
        <mi>f</mi>
        <mo>:</mo>
        <mi>z</mi>
      </mrow>
      <mo>=</mo>
      <mrow>
        <mi>sin</mi>
        <mrow>
          <mo>(</mo>
            <msqrt>
              <mrow>
                <msup>
                  <mi>x</mi>
                  <mn>2</mn>
                </msup>
                <mo>+</mo>
                <msup>
                  <mi>y</mi>
                  <mn>2</mn>
                </msup>
              </mrow>
            </msqrt>
          <mo>)</mo>
        </mrow>
      </mrow>
    </math> je </p>

```

```
<msp:evaluate>
  MSPLive3D[graf]
</msp:evaluate>
...
```

Aj napriek tomu, že všetky nastavenie na stránke sú správne, text sa nezobrazí korektne.

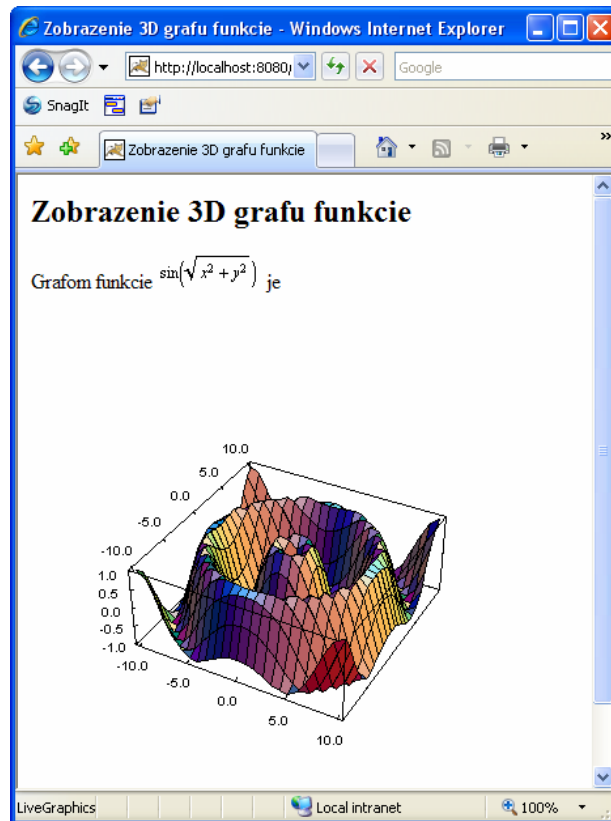


Momentálne jedinou možnosťou, o ktorej vieme, je použiť zobrazenie matematického textu vo forme obrázku napríklad takto:

```
<msp:evaluate>
  graf = Plot3D[Sin[Sqrt[x^2 + y^2]], {x, -10, 10}, {y, -10, 10}];
</msp:evaluate>

  <p> Grafom funkcie
  <msp:evaluate>
    MSPFormat[Sin[Sqrt[x^2 + y^2]], TraditionalForm]
  </msp:evaluate> je </p>

<msp:evaluate>
  MSPLive3D[graf]
</msp:evaluate>
```



Na ukážke vidíme, že obrázok nie je správne umiestnený, ale jeho umiestnenie (vertikálne zarovnanie s ostatným textom) nedokážeme ovplyvniť.

Príkaz MSPLive3D môžeme použiť na zobrazenie ľubovoľných objektov typu **-Surface Graphics-**, alebo objekt typu **-Graphics3D-**. Popísaným spôsobom môžeme zobrazovať aj výsledok príkazov **ListPlot3D**, **ParametricPlot3D** a pod.

3 Dynamické stránky po prvý raz

V tejto sekcii si ukážeme, ako je možné v prostredí web**MATHEMATICE** pracovať s premennými, ktoré „prevezmeme“ od užívateľa webovej stránky. Jediný spôsob, ako je možné takéto premenné prevziať je prostredníctvom HTML formulárov. HTML formulárom sme sa venovali v samostatnej kapitole, preto teraz zopakujeme len základy. Týmto spôsobom budeme po prvý krát vytvárať dynamické webové stránky.

Príklad 6

Vytvorte jednoduchý HTML formulár, ktorého úlohou je prevziať od užívateľa stránky obsah textovej premennej a túto premennú následne zobrazit'.

Pozrieme sa na jednoduchý zdrojový dokument takéhoto formulára:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
href="/webMathematica/Resources/XSL/mathml.xsl"?>

<%@ page language="java" %>
<%@ page contentType="text/html; charset=utf-8" %>
<%@ taglib uri="/webMathematica-taglib" prefix="msp" %>

<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta name="author" content="Monika Kovacova"/>
    <meta http-equiv="Content-type" content="text/html;
charset=utf-8" />
    <title>Premené</title>
  </head>

  <body bgcolor="#ffffff">
<h2>Premenné</h2>

<msp:allocateKernel>
<msp:evaluate>
  MSPPageOptions[ "ContentType" -> "text/xml"];
</msp:evaluate>

  <form action="priklad06.jsp" method="post">

    <p>Zadaj textovú premennú:
<input type="text" name="prem1" size="10" />
    </p>

    Hodnota premennej je:
    <msp:evaluate>
    $$prem1
    </msp:evaluate>

    <br/><br/>

    <input type="submit" name="tlacidlo" value="Odoslat" />

  </form>

</msp:allocateKernel>

</body>
</html>
```

Podstatné časti toho dokumentu sme vyznačili tučným typom písma.

Na stránke sme použili formulár s názvom **priklad06.jsp**. Po odoslaní formulára (stlačením tlačidla „Odoslat“) bude na server zaslaná požiadavka zobrazit' stránku `priklad06.jsp` –

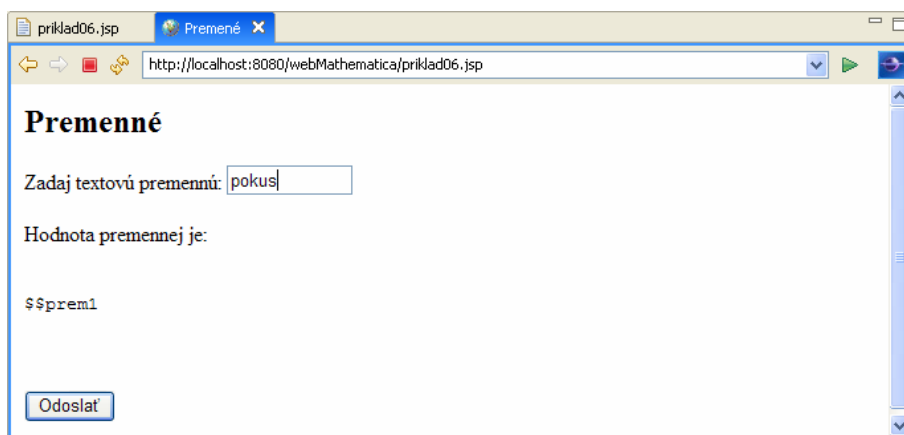
pretože to je názov definovaný v tagu `form`. Forma zaslania bola zvolená „post“. Odporúčame vám používať túto metódu pre zasielanie v rámci formulárov.

V zdrojovom dokumente ďalej nájdete tag `<input type="text" value="" name="prem1">`, ktorý umožní užívateľovi web stránky zadať textovú hodnotu, ktorá sa následne pri ďalšom prechode formulárom zapíše do HTML premennej `prem1`. Pri prvom prechode stránky textová premenná `prem1` ešte nenadobudla žiadnu hodnotu, preto je okienko prázdne. Ak chceme použiť hodnotu HTML premennej `prem1` vo výpočtovom prostredí *MATHEMATICE*, používame ju ako `$$prem1`. Pretože pri prvom zobrazení formuláru HTML premenná `prem1` nenadobudla žiadnu hodnotu, ani web*MATHEMATICA* premenná `$$prem1` nemá priradenú žiadnu hodnotu.

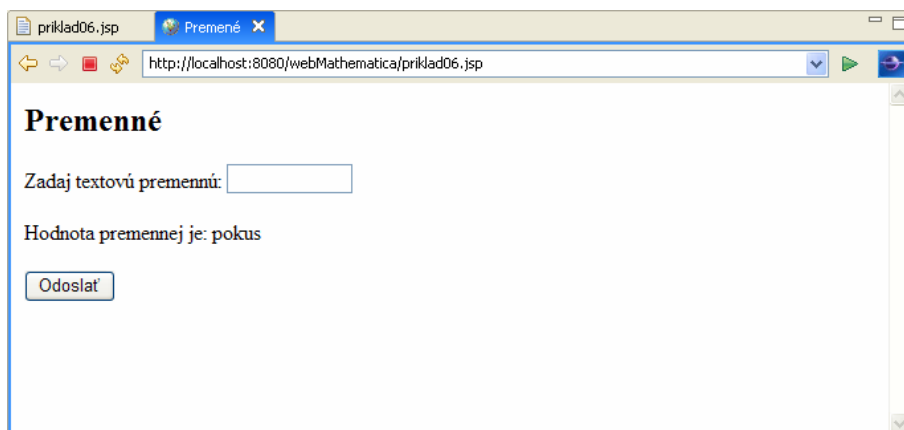
Stránka sa zobrazí takto:



Keď užívateľ zadá hodnotu premennej (do okienka) stránka sa zobrazí v nasledujúcom tvare.



Po stlačení tlačidla „Odoslať“ sa zobrazí tento výsledok.

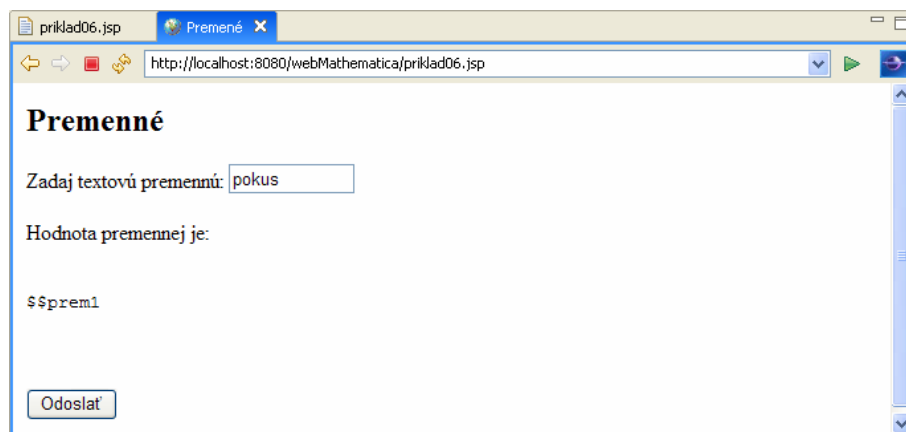


Podme teraz podrobnejšie analyzovať, čo sa udialo pri použití tejto stránky. Mechanizmus odovzdávania hodnoty medzi HTML premennou, ktorá svoju hodnotu nadobudne v rámci formulára a premennou systému webMATHEMATICA je veľmi dôležité pochopiť, pretože v takmer všetkých dynamických stránkach sa tento mechanizmus používa.

HTML premenná môže svoju hodnotu získať buď ako default hodnotu

```
<input type="text" name="prem1" size="10" value="pokus" />
```

alebo prostredníctvom zadania tejto hodnoty užívateľom stránky. Aby sme predišli možným problémom, pri pokuse o spracovanie formulára, keď premenné formulára nenadobudnú žiadne hodnoty, je obvyklé pre každú formulárovú premennú takúto default hodnotu definovať. Ak je takáto default hodnota definovaná, už pri prvom zobrazení stránky sa v príslušnom okienku zobrazí.



Užívateľ samozrejme môže jej hodnotu kedykoľvek zmeniť. Dôležité je ale uvedomiť si, že HTML je bez-stavový protokol. Inými slovami – nikto nikde nezaznamenáva históriu premenných použitých vo formulári a ak nie je táto otázka špeciálnym spôsobom v zdrojovom kóde ošetrená, predchádzajúce hodnoty formulárových premenných nie je možné dohľadať (prípadne znovu zobraziť).

Ak chceme obsah HTML premennej používať vo výpočtoch musíme jej hodnotu čo najrýchlejšie priradiť premenným, ktorých obsah webMATHEMATICA na serveri uchováva. Ak budeme pracovať len s dvojicou premenných `prem1` a `$$prem1`, budeme stále bojovať so skutočnosťou, že premenná `$$prem1` **nadobudne svoju hodnotu len v následnom kroku po prechode formulárom**. Odporúčame na vytvorenej stránke `priklad06.jsp` správanie formulára otestovať.

Z programátorského hľadiska najsprávnejšie je hneď po získaní hodnoty premennej z HTML formulára, otestovať, či nadobudla prípustnú hodnotu a priradiť jej hodnotu premennej systému webMATHEMATICA, ktorá už nie je citlivá na viaceré prechody formulárom.

```
<form action="priklad06.jsp" method="post">
```

```
<p>Zadaj textovú premennú:
```

```
<input type="text" name="prem1" size="10" value="pokus"/>
</p>
```

```
Hodnota premennej je:
```

```
<msp:evaluate>
```

```
If [MSPValueQ[$$prem1],
```

```
    prem1 = $$prem1, prem1=ToString["moja hodnota"]];
```

```
prem1
```

```
</msp:evaluate>
```

```
<br/><br/>
```

```
<input type="submit" name="tlacidlo" value="Odoslat" />
```

```
</form>
```

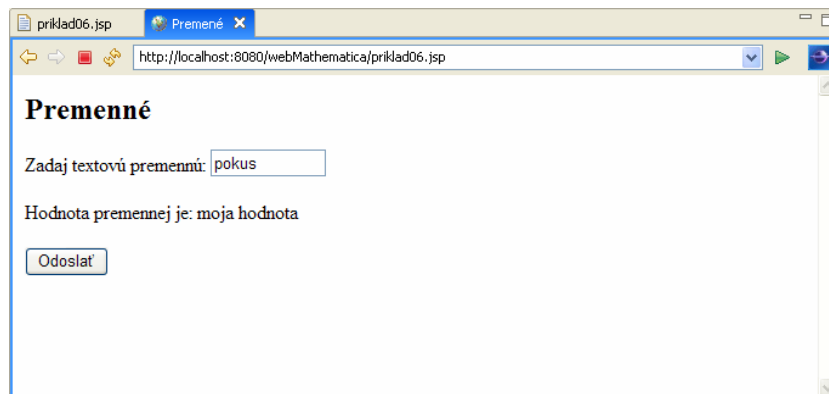
Zdrojový kód tohto príkladu sme zmenili pridaním testovacej podmienky **if**, v ktorej testujeme, či premenná **\$\$prem1** nadobudla hodnotu. Použili sme nový príkaz **webMATHEMATICE**.

MSPValueQ[\$\$premenna]

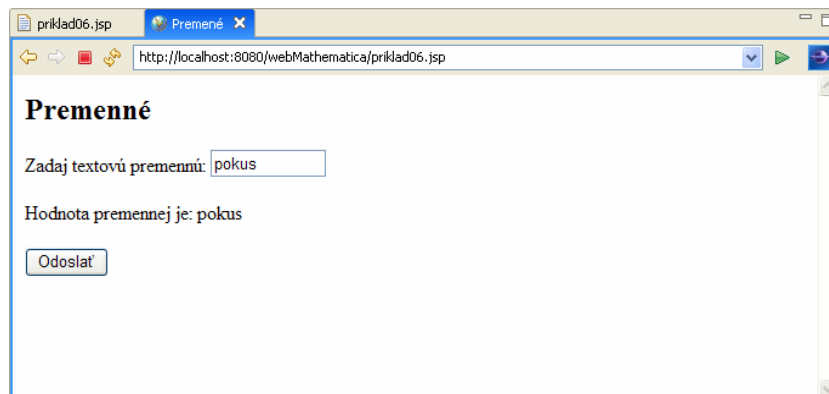
Príkaz **MSPvalueQ** je testovací príkaz a nadobudne hodnotu **True**, ak premená **\$\$premenna** nadobudla nejakú hodnotu (akúkoľvek). Ak premenná **\$\$premenna** má hodnotu **Null** (teda zatiaľ nenadobudla žiadnu hodnotu), výsledkom testovacej funkcie je hodnota **False**.

Pomocou tohto testovacieho príkazu v našom príklade overíme, či premenná **\$\$prem1** nadobudla nejakú hodnotu. Ak nie, tak novej premennej (premennej **MATHEMATICE**) s názvom **prem1** jej priradíme textový reťazec **["moja hodnota"]**. Ak premenná **\$\$prem1** nadobudla nejakú hodnotu, tak do novej premennej (premennej **MATHEMATICE**) s názvom **prem1** priradíme hodnotu, ktorú získala prostredníctvom HTML formulára.

Pri prvom prechode formulárom premenná **\$\$prem1** nenadobudla žiadnu hodnotu, preto sa zobrazí text **moja hodnota**.



Všimnite si, že v okienku formulára je zobrazená default hodnota HTML premennej. Pretože ale hodnota tejto HTML premennej ešte nebola odovzdaná do prostredia **webMATHEMATICE**, premenná **\$\$prem1** nadobudla hodnotu **moja hodnota**. Pri ďalšom prechode formulárom už HTML premenná svoju hodnotu odovzdá a tak sa aj hodnota premennej **\$\$prem1** zmení. Následne sa vďaka testovaciemu príkazu zmení aj hodnota premennej **prem1** z prostredia **MATHEMATICE**.



Ešte raz si podrobne vysvetlime s akými rôznymi typmi premenných v rámci formulára pracujeme:

HTML premenná – prem1

```
<input type="text" name="prem1" size="10" value="pokus"/>
```

Tento typ premennej je bez-stavový, po každom zobrazení stránky nadobudne buď hodnotu zadanú na stránke, alebo default hodnotu definovanú v parametri `value`.

webMATEHMATICA premenná

```
$$prem1
```

Po **každom** odoslaní formulára (t.j. v nasledujúcom prechode stránkou) prevezme automaticky hodnotu od HTML premennej. Ak HTML premenná nenadobudla žiadnu hodnotu, tak webMATEHMATICA premenná nadobudne automaticky hodnotu `Null`.

MATHEMATICA premenná

`prem1` – ale len v prostredí ohraničenom tagmi `<msp:evaluate>` a `</msp:evaluate>`

Táto premenná si svoju hodnotu uchová aj pri viacnásobnom prechode formulárom. Hodnotu ale nenadobúda automaticky. Musíme v rámci tvorby zdrojového kódu stránky zabezpečiť, aby nejakú hodnotu nadobudla. Tento typ premennej má len obmedzenú oblasť existencie. Mimo prostredia ohraničeného tagmi `<msp:evaluate>` a `</msp:evaluate>`, nemôžeme túto premennú žiadnym spôsobom použiť, ani jej hodnotu priradiť nejakej inej premennej. Ide o typickú lokálnu premennú.

V tomto príklade sme sa venovali len hypotetickým otázkam odovzdávania hodnôt v rámci formulára. Prvý jednoduchý reálny formulár zostrojíme v nasledujúcom príklade.

Príklad 7

Vytvorte jednoduchý HTML formulár, ktorého úlohou je sčítať dve čísla zadané užívateľom.

Pozrieme sa na jednoduchý zdrojový dokument takéhoto formulára. Prvý pokus o vytvorenie formulára nebude ten najefektívnejší možný, ale pretože našou snahou je vysvetliť podstatu programovania vo webMATEHMATICE, budeme postupovať od neefektívneho spôsobu ku „čistému“ programátorskému kódu.

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
href="/webMathematica/Resources/XSL/mathml.xsl"?>

<%@ page language="java" %>
<%@ page contentType="text/html; charset=utf-8" %>
<%@ taglib uri="/webMathematica-taglib" prefix="msp" %>

<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta name="author" content="Monika Kovacova"/>
    <meta http-equiv="Content-type" content="text/html;
charset=utf-8" />
    <title>Formulár na sčítanie dvoch čísel</title>
  </head>
```

```

<body bgcolor="#ffffff">
<h2>Formulár na sčítanie dvoch čísel</h2>

<msp:allocateKernel>

    <form action="priklad07.jsp" method="post">

        <p>Zadaj prvé číslo:
        <input type="text" name="cislo1" size="10" value="2" />
        </p>

        <p>Zadaj druhé číslo:
        <input type="text" name="cislo2" size="10" value="4"/>
        </p>

        <p>Súčtom čísel
        <msp:evaluate> $$cislo1 </msp:evaluate> a
        <msp:evaluate> $$cislo2 </msp:evaluate> je
        <msp:evaluate> $$cislo1 + $$cislo2</msp:evaluate>
        </p>

        <input type="submit" name="tlacidlo" value="Odoslat" />

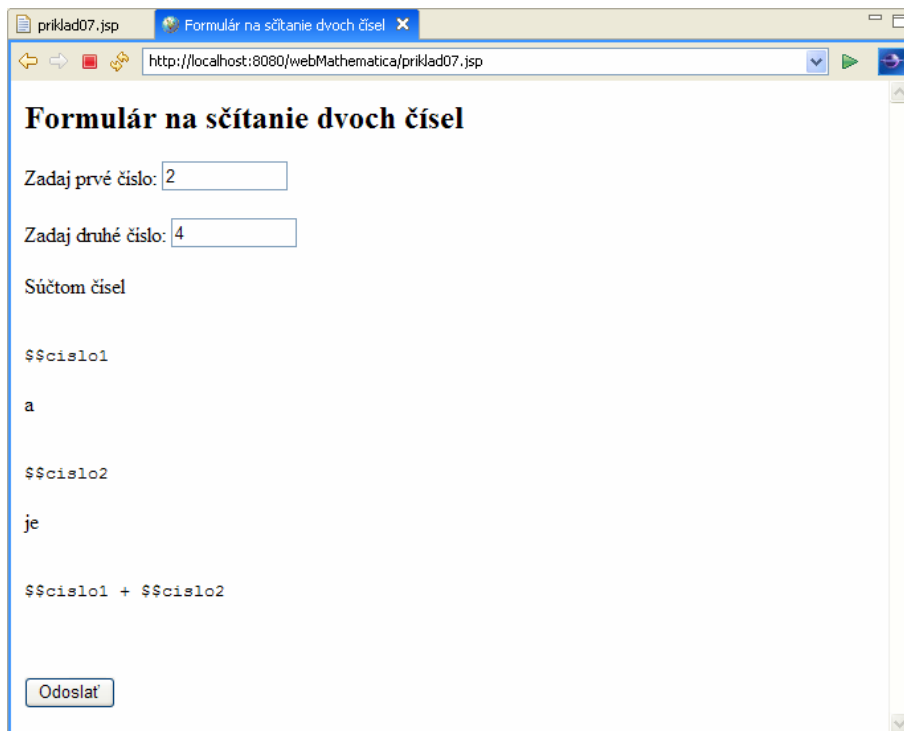
    </form>

</msp:allocateKernel>

</body>
</html>

```

Pri prvom zobrazení stránky výsledok vyzerá takto:



S výsledkom nemôžeme byť spokojní a preto sa pokúsime kód jsp stránky opraviť.

Pretože ide o veľmi jednoduchý formulár môžeme použiť príkaz `webMATHEMATICE – MSPBlock`. Tento príkaz umožňuje pracovať priamo s prevzatými premennými `webMATHEMATICE` – s premennými typu `$$prem`. Tento spôsob je jedným z najjednoduchších spôsobom práce s premennými vo formulároch, ale dôsledkom „jednoduchého použitia“ je zároveň obmedzenia jeho použitia len na „jednoduché“ formuláre. V nasledujúcom príklade si ukážeme aj trochu všeobecnejší spôsob, ktorý je z programátorského hľadiska správnejší a predovšetkým umožňuje všeobecné použitie.

Vráťme sa teraz ku nášmu príkladu. Pozrite sa, ako sme zdrojový kód príkladu upravili

```
<form action="priklad07A.jsp" method="post">

<p>Zadaj prvé číslo:
<input type="text" name="cislo1" size="10" value="2"/>
</p>

<p>Zadaj druhé číslo:
<input type="text" name="cislo2" size="10" value="4"/>
</p>

<p>Súčtom čísel je
<msp:evaluate>
    MSPBlock[{$$cislo1, $$cislo2},
        MSPFormat[{$$cislo1+$cislo2,MathMLForm]
    ]
</msp:evaluate>
</p>

<br/><br/>
<input type="submit" name="tlacidlo" value="Odoslať" />

</form>
```

Príkaz `MSPBlock` má podobnú funkciu ako príkaz `Block` v `MATHEMATICE`. Jeho štruktúra je nasledovná:

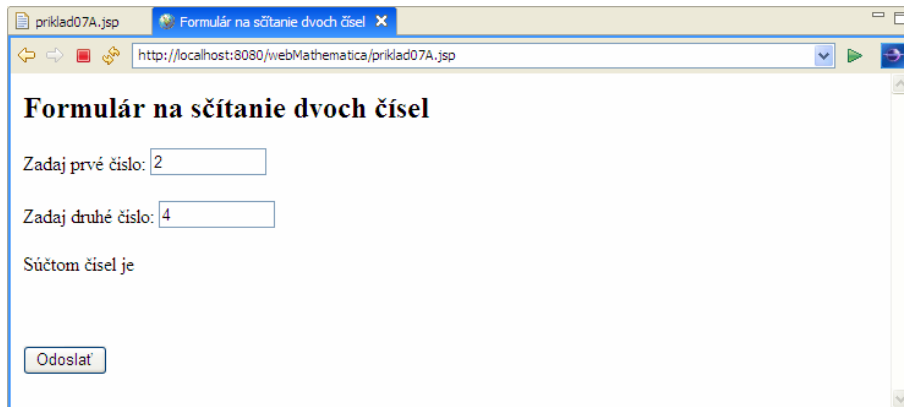
```
MSPBlock[
    {$$lokalnapremenna1, $$lokalna premenna2,...},
    telo procedúry (príkazy oddelené bodkočiarkami);
    prikaz 2;
    prikaz 3;
    prikaz 4
]
```

Najskôr do krútených zátvoriek uvedieme **zoznam všetkých `webMATHEMATICA` premenných, ktoré sme načítali prostredníctvom formulára a chceme v rámci tohto príkazu spracovať**. Pri použití tohto príkazu môžeme premenné použiť priamo a nemusíme sa starať o ich typ, charakter, alebo obsah.

Ak lokálne premenné nenadobudli zatiaľ svoj obsah, svoju hodnotu – napríklad pri prvom prechode stránkou, v rámci príkazu `MSPBlock` sa s nimi pracuje ako keby nadobudli hodnotu `Null`. Preto pri prvom prechode stránkou nevidíme žiaden výsledok – a to aj v prípade, že sme vo vstupných políčkach premenným už nejakú hodnotu pomocou parametra `value='....'` v tagu `<input>` už priradili (pozrite sa na predchádzajúci zdrojový kód).

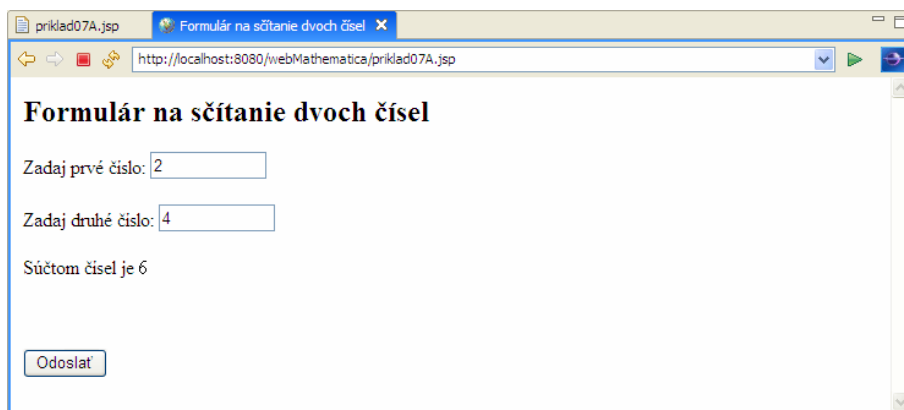
Po odoslaní formulára sú hodnoty všetkých premenných uvedených v zozname lokálnych premenných v príkaze `MSPBlock` interpretované výpočtovým jadrom systému *MATHEMATICA*. Ak je interpretácia úspešná, výsledok je zobrazený na webovej stránke. Výsledkom výpočtu je v tomto prípade výsledok, ktorý získame realizáciou postupnosti príkazov, ktoré sú uvedené v tele príkazu `MSPBlock`.

Pri prvom prechode formulárom, ešte premenné `$$cislo1` a `$$cislo2` nenadobudli žiadnu hodnotu



The screenshot shows a web browser window with the address bar displaying `http://localhost:8080/webMathematica/priklad07A.jsp`. The page title is "Formulár na sčítanie dvoch čísel". The form contains two input fields: "Zadaj prvé číslo:" with the value "2" and "Zadaj druhé číslo:" with the value "4". Below the fields, the text "Súčtom čísel je" is displayed. At the bottom left, there is a button labeled "Odoslať".

Pri druhom prechode formulárom už hodnotu nadobudli a ich hodnota bola interpretovaná a použitá ako súčasť výpočtu.



The screenshot shows the same web browser window as above. The form is identical, but the text "Súčtom čísel je" is now followed by the number "6", indicating the result of the calculation. The button "Odoslať" is still present at the bottom left.

Tento postup je veľmi jednoduchý a elegantný, ale **jeho použitie je obmedzené**. Prvým problémom je, že formulár na jsp stránke si nedokáže zapamätať hodnotu premenných, ktoré boli do vstupných formulárových políček zapísané. Dôvodom je, že HTML protokol je bezstavový.



The screenshot shows the same web browser window as above. The form is identical, but the first input field "Zadaj prvé číslo:" now contains the value "3", while the second field "Zadaj druhé číslo:" remains "4". The text "Súčtom čísel je" is still followed by the number "6". The button "Odoslať" is still present at the bottom left.

Ak napríklad v našom formulári zmeníme vstupné údaje – ako na predchádzajúcom obrázku a odošleme na spracovanie dostaneme takúto výslednú stránku. Stránka síce vypočítala súčet čísel správne, ale pretože nikde nie je uvedené, **aký príklad sme počítali**, môže (najmä v prípade študentov) prísť ku chybnjej interpretácii výsledku.



V tomto jednoduchom príklade sa zdá byť táto skutočnosť nepodstatná, ale pri zložitejších príkladoch môže byť takto zobrazená stránka veľmi zavádzajúca.

Tento postup nemôžeme použiť ani v situáciách, keď chceme do vstupného políčka zadať zložitejší vstup napr. pri načítaní súborov, zadání predpisu funkcie, alebo pri použití chybnjej syntaxe vstupných premenných (napr. $x[. . 1]$). Pretože hodnota premennej, ktorá je zadaná do vstupného políčka je priamo interpretovaná, môže interpretácia chybného vstupu spôsobiť pád celého výpočtového jadra. V prípadoch, keď sa nedá príliš spoliehať na znalosti syntaxe programového systému *MATHEMATICA* u užívateľa webovej stránky, je preto výhodnejšie použiť postup popísaný v nasledujúcom príklade. Je síce zložitejší, ale predstavuje robustnejší spôsob vytvárania jsp aplikácií.

Príklad 7 – po druhý raz

Vytvorte jednoduchý HTML formulár, ktorého úlohou je sčítať dve čísla zadané užívateľom.

V tomto príklade budeme riešiť rovnakú úlohu, ako v predchádzajúcom príklade, ale postup, ktorý si zvolíme bude predstavovať robustnejší, „blbuvzdornejší“ spôsob vytvárania jsp stránok.

Prvý pokus o vytvorenie formulára nebude ten najefektívnejší možný, ale pretože našou snahou je vysvetliť podstatu programovania vo *webMATHEMATICE*, budeme postupovať od neefektívneho spôsobu ku „čistému“ programátorskému kódu.

Začneme rovnakým zdrojovým kódom ako v predchádzajúcom príklade. Pri prvom zobrazení stránky dostaneme výsledok zobrazený na vedľajšom obrázku.



Podme porovnať zobrazený výsledok so zdrojovým kódom `priklad07.jsp` stránky.

```
<form action="priklad08.jsp" method="post">

<p>Zadaj prvé číslo:
<input type="text" name="cislo1" size="10" value="2" />
</p>

<p>Zadaj druhé číslo:
<input type="text" name="cislo2" size="10" value="4"/>
</p>

<p>Súčtom čísel
<msp:evaluate> $$cislo1 </msp:evaluate> a
<msp:evaluate> $$cislo2 </msp:evaluate> je
<msp:evaluate> $$cislo1 + $$cislo2</msp:evaluate>
</p>

<input type="submit" name="tlacidlo" value="Odoslat" />

</form>
```

Najskôr sa na stránke zobrazia dve formulárové políčka, v ktorých je umiestnená default hodnota HTML premenných `cislo1` a `cislo2`. Pretože ide o prvý prechod formulárom, HTML premenné ešte nemali možnosť odovzdať svoju hodnotu premenným `$$cislo1` a `$$cislo2`. V nasledujúcej časti stránky sa preto zobrazia texty `$$cislo1` a `$$cislo2`. Tak ako v predchádzajúcej verzii tohto príkladu musíme skonštatovať, že takto by stránka nemala vyzeráť, ale tento problém odstránime za chvíľu. V tomto príklade sme zámerne ponechali tento spôsob kódovania, aby sme podrobne vysvetlili spôsob odovzdávania si hodnôt medzi jednotlivými premennými.

Po odoslaní formulára (žiadne hodnoty sme vo vstupných políčkach nemenili) sa stránka zobrazí takto:



Po stlačení tlačidla „**Odoslat**“ sme danú stránku zobrazili druhý krát a HTML premenné `cislo1` a `cislo2` odovzdali svoju hodnotu premenným `$$cislo1` a `$$cislo2`. Výsledok sme

ale ešte stále nezískali. Teraz pristúpme ku vysvetleniu, prečo sa na stránke nezobrazil výsledok výpočtu – číslo 6 – aj napriek tomu, že všetky premenné svoje hodnoty nadobudli.

HTML premenná je textová premenná. Táto premenná svoju hodnotu po prechode formulárom síce odovzdá premennej `$$cislo1`, ale opäť **len ako text**. Ani v *MATHEMATICE* nie je možné sčítať dve textové premenné. Ak chceme s obsahom týchto premenných manipulovať – napríklad sčítať ich hodnoty, musíme ich najskôr transformovať do tvaru výrazu. Najčastejšie sa používa príkaz

ToExpression[premenna]

Upravme časť zdrojového dokumentu a pozrime sa teraz na výsledok

```
<p>Súčtom čísel  
<msp:evaluate> $$cislo1 </msp:evaluate> a  
<msp:evaluate> $$cislo2 </msp:evaluate> je  
<msp:evaluate>  
    ToExpression[$$cislo1] + ToExpression[$$cislo2]  
</msp:evaluate>  
</p>
```

Stránka po odoslaní formulára zobrazí už správny výsledok.



Na záver príkladu potrebujeme ešte ukázať, ako je možné zabezpečiť, aby sa stránka aj pri prvom prechode formulárom zobrazila korektne. Ukážeme si efektívny spôsob odovzdávania hodnôt premenných.

```
<form action="priklad07.jsp" method="post">  
  
<p>Zadaj prvé číslo:  
<input type="text" name="cislo1" size="10" value="2" />  
<msp:evaluate>  
    If[MSPValueQ[$$cislo1],  
        cislo1 = ToExpression[$$cislo1], cislo1 = ToExpression["2"]];  
</msp:evaluate>  
</p>
```

```

<p>Zadaj druhé číslo:
<input type="text" name="cislo2" size="10" value="4"/>
<msp:evaluate>
If[MSPValueQ[$$cislo2],
      cislo2 = ToExpression[$$cislo2], cislo2 = ToExpression["4"]];
</msp:evaluate>
</p>

<p>Súčtom čísel
<msp:evaluate>
      MSPFormat[cislo1, MathMLForm] <>" a " <>
      MSPFormat[cislo2, MathMLForm] <>" je " <>
      MSPFormat[cislo1+cislo2, MathMLForm]
</msp:evaluate>
</p>

<input type="submit" name="tlacidlo" value="Odoslat" />

</form>

```

V tejto verzii formulára sme ošetrili zobrazovanie premenných hneď pri prvom prechode formulárom. použili sme funkciu **MSPValueQ**. Ak premenná **cislo1** nadobudne nejakú hodnotu (t.j. pri druhom zobrazení stránky), tak jej hodnotu zmeníme z textovej hodnoty na číselnú a hneď ju priradíme *MATHEMATICA* premennej **cislo1**. Ak premenná **cislo1** nenadobudla žiadnu hodnotu (t.j. pri prvom prechode formulárom), tak *MATHEMATICA* premennej **cislo1** priradíme default číselnú hodnotu. Rovnakú operáciu urobíme aj pre premennú **cislo2**. V ďalších výpočtoch pracujeme už len s premennými **cislo1** a **cislo2** vo výpočtovom prostredí *MATHEMATICE* v rámci tejto stránky.

V opravenej verzii stránky zabezpečíme krajší spôsob zapísania získaných výsledkov. Použili sme príkaz **MSPFormat** a spájanie reťazcov.

Takto upravená stránka sa pri prvom zobrazení aj po odoslaní formulára na spracovanie tvári konzistentne a vyzerá takto:



Poslednú chybu v správaní stránky zistíme, keď začneme stránku testovať. Ak zmeníme hodnoty vo formulárových poličkách napríklad na 3 a 8 a stránku odošleme na spracovanie, vo výstupe získame síce správny výsledok ale vo formulárových poličkach sa nám zobrazia

opäť pôvodné hodnoty HTML premenných `cislo1` a `cislo2`. Správanie stránky vidíte aj na nasledujúcom obrázku.



Toto správanie je nekorektné a môže spôsobovať problémy, najmä ak do formulárového políčka (prípadne do viacerých políčok) potrebujeme zapísať zložitejší výraz. Chyba v jednom zo zápisov nás donúti zadávať celý obsah vstupných políčok znovu. Druhým problémom je skutočnosť, že po odoslaní stránky nikde nemáme zobrazené, čo sme skutočne do formulárového políčka zadali.

Túto chybu môžeme odstrániť pomocou príkazu

MSPValue[*premenna*, *default hodnota*]

Ak **premenna** nadobudla nejakú hodnotu tak, jej hodnotu zachová. V opačnom prípade jej priradí default hodnotu definovanú v druhej časti tohto príkazu.

Opravme zdrojový kód

```
<form action="priklad07.jsp" method="post">

<p>Zadaj prvé číslo:
<input type="text" name="cislo1" size="10"
      value="<msp:evaluate>MSPValue[ $\$$  $\$$ cislo1,"2"]</msp:evaluate>" />
<msp:evaluate>
If[MSPValueQ[ $\$$  $\$$ cislo1],
    cislo1 = ToExpression[ $\$$  $\$$ cislo1], cislo1 = ToExpression["2"];
</msp:evaluate>
</p>

<p>Zadaj druhé číslo:
<input type="text" name="cislo2" size="10"
      value="<msp:evaluate>MSPValue[ $\$$  $\$$ cislo2,"4"]</msp:evaluate>" />
<msp:evaluate>
If[MSPValueQ[ $\$$  $\$$ cislo2],
    cislo2 = ToExpression[ $\$$  $\$$ cislo2], cislo2 = ToExpression["4"];
</msp:evaluate>
</p>

<p>Súčtom čísel
<msp:evaluate>
    MSPFormat[cislo1, MathMLForm] <>" a " <>
    MSPFormat[cislo2, MathMLForm] <>" je " <>
</msp:evaluate>
```

```

        MSPFormat[cislo1+cislo2, MathMLForm]
    </msp:evaluate>
</p>

<input type="submit" name="tlacidlo" value="Odoslať" />

</form>

```

Zmeny sme vyznačili tučným typom písma. Pri prvom zobrazení sa stránka zobrazí v tvare:



Ak zmeníme hodnoty vstupných formulárových políček dostaneme opäť korektnú stránku



Vráťme sa na chvíľu ešte ku prvej verzii tohto príkladu, v ktorej sme zabezpečili interpretáciu príkazov pomocou príkazu **MSPBlock**. Ak zdrojový kód tejto verzie príkladu doplníme o príkazy **MSPValue** a testovanie hodnoty premennej **MSPValueQ**, vyriešime problém so „zapamätaním si hodnoty premenných“. Nasleduje upravený zdrojový kód stránky.

```

<form action="priklad07A.jsp" method="post">

<p>Zadaj prvé číslo:
<input type="text" name="cislo1" size="10"
        value="<msp:evaluate>MSPValue[ $\$$  $\$$ cislo1,"2"]</msp:evaluate>" />
<msp:evaluate>

```

```

If[MSPValueQ[ $$$cislo1$ ],
     $cislo1 = ToExpression[ $$$cislo1$ ], cislo1 = ToExpression["2"]];
</msp:evaluate>
</p>

<p>Zadaj druhé číslo:
<input type="text" name="cislo2" size="10"
    value="<msp:evaluate>MSPValue[ $$$cislo2$ ,"4"]</msp:evaluate>" />
<msp:evaluate>
If[MSPValueQ[ $$$cislo2$ ],
     $cislo2 = ToExpression[ $$$cislo2$ ], cislo2 = ToExpression["4"]];
</msp:evaluate>
</p>

<p>Súčtom čísel je
<msp:evaluate>
    MSPBlock[{{ $$$cislo1$ ,  $$$cislo2$ },
        MSPFormat[ $$$cislo1+$$cislo2$ ,MathMLForm]
    ]
</msp:evaluate>
</p>

<br/><br/>
<input type="submit" name="tlacidlo" value="Odoslať" />

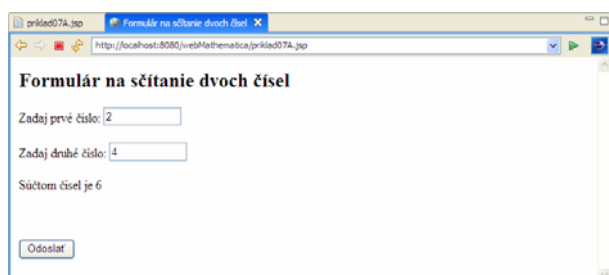
</form>$$ 
```

Výsledok takto upravenej stránky je:

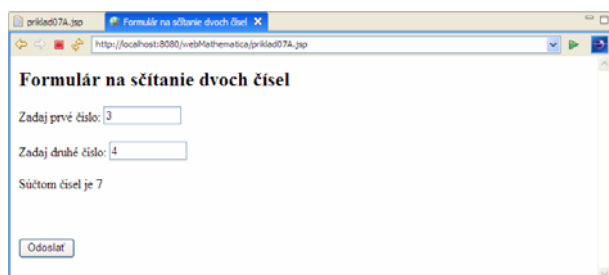
- prvé zobrazenie stránky



- po odoslaní formulára



- po zadaní vlastných vstupných hodnôt a odoslaní formulára na spracovanie.



Je ťažké zhodnotiť, ktorý z navrhovaných spôsobov práce s premennými v rámci web*MATHEMATICE* je efektívnejší. Prvý spôsob, pomocou príkazu `MSBlock`, je jednoduchší, vyžaduje kratší a jednoduchší zdrojový kód. Druhý spôsob je univerzálnejší a pretože ukladá získané HTML premenné do *MATHEMATICA* premenných, umožňuje väčšiu variabilitu.