

# Aplety



A0B36PR2-Programování 2  
Fakulta elektrotechnická  
České vysoké učení technické

# Exkurse do principu webové aplikace

Předpoklad - propojení programu klient (browser) a server

1. krok: zadání URL (adresy) serveru v klientském programu a odeslání zprávy
2. krok: server vygeneruje HTML stránku a zašle ji klientovi, spojení končí

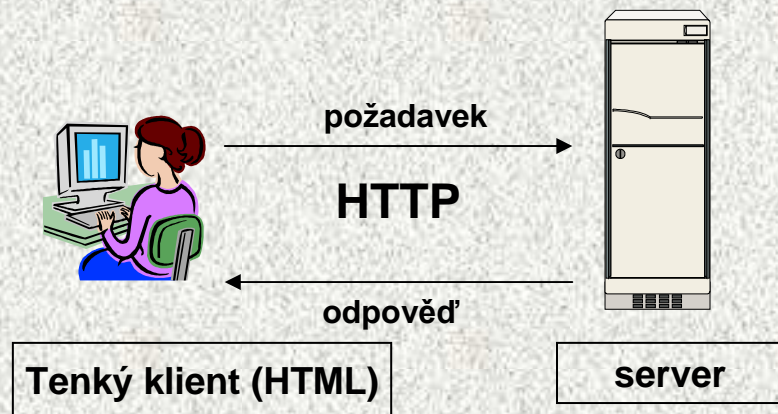
HTML stránka klienta obsahuje možnosti pro komunikaci:

1. nabídku URL zdrojů (serverů) k opětné komunikaci
2. možnost zadání vstupních dat pro server
3. funkce odešli (určenému serveru – URL), odešle zprávu protokolem HTTP

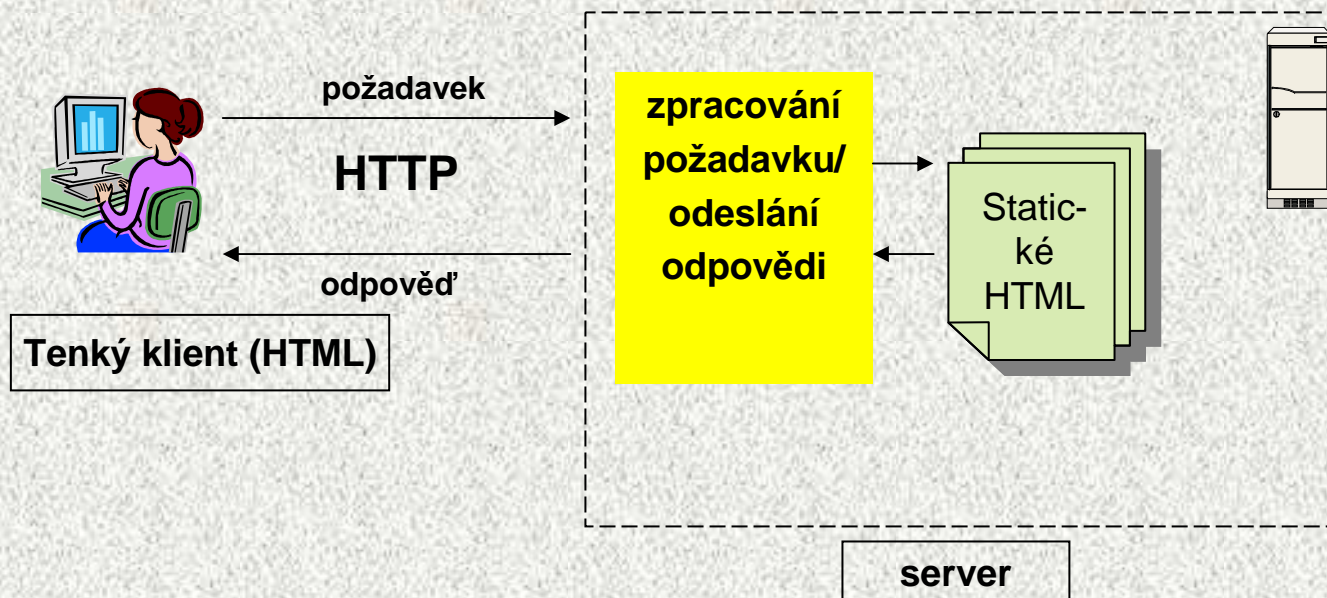
Zpráva zaslaná klientem obsahuje:

1. data zadaná uživatelem
  - parametry + hodnoty bývají zadávány formou vyplnění formuláře
  - hodnoty dat bývají zpracovány, kontrolovány a zobrazeny před odesláním (JavaScript, Java Applet apod.)
2. požadovaný způsob zpracování – serverový program, emailová adresa, ...
  - bývá to program v CGI, SSJS, ASP, ale nejčastěji skript v PHP

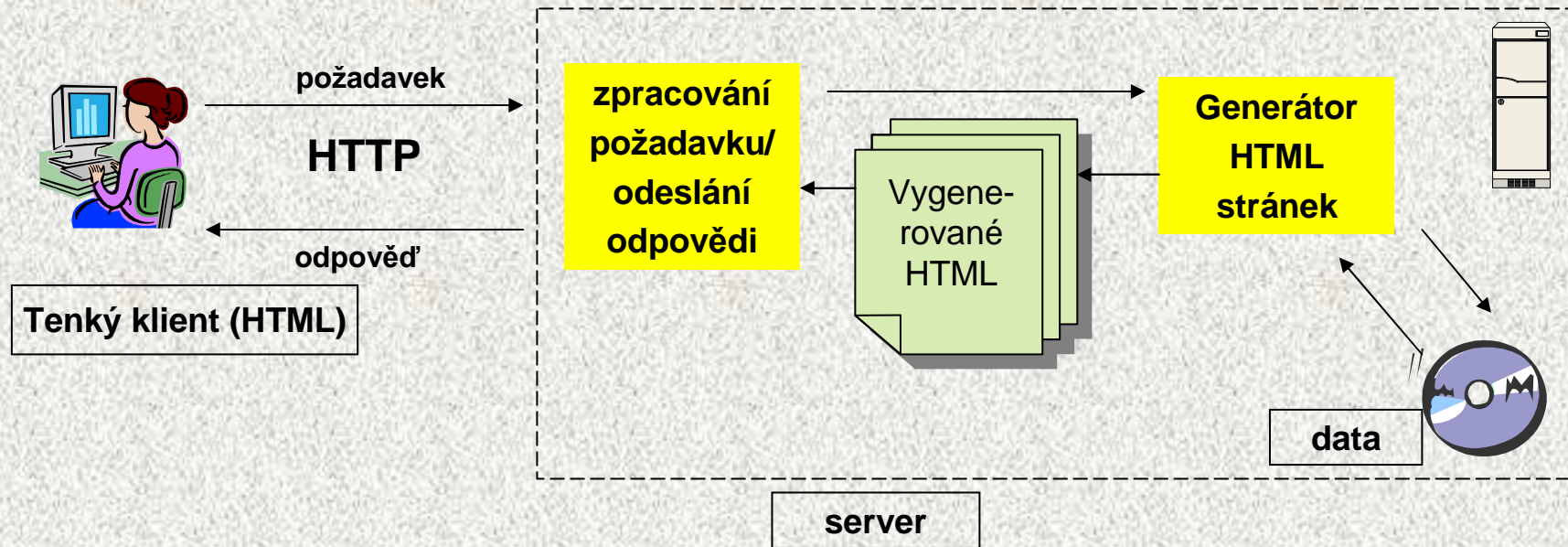
# Architektura web aplikace



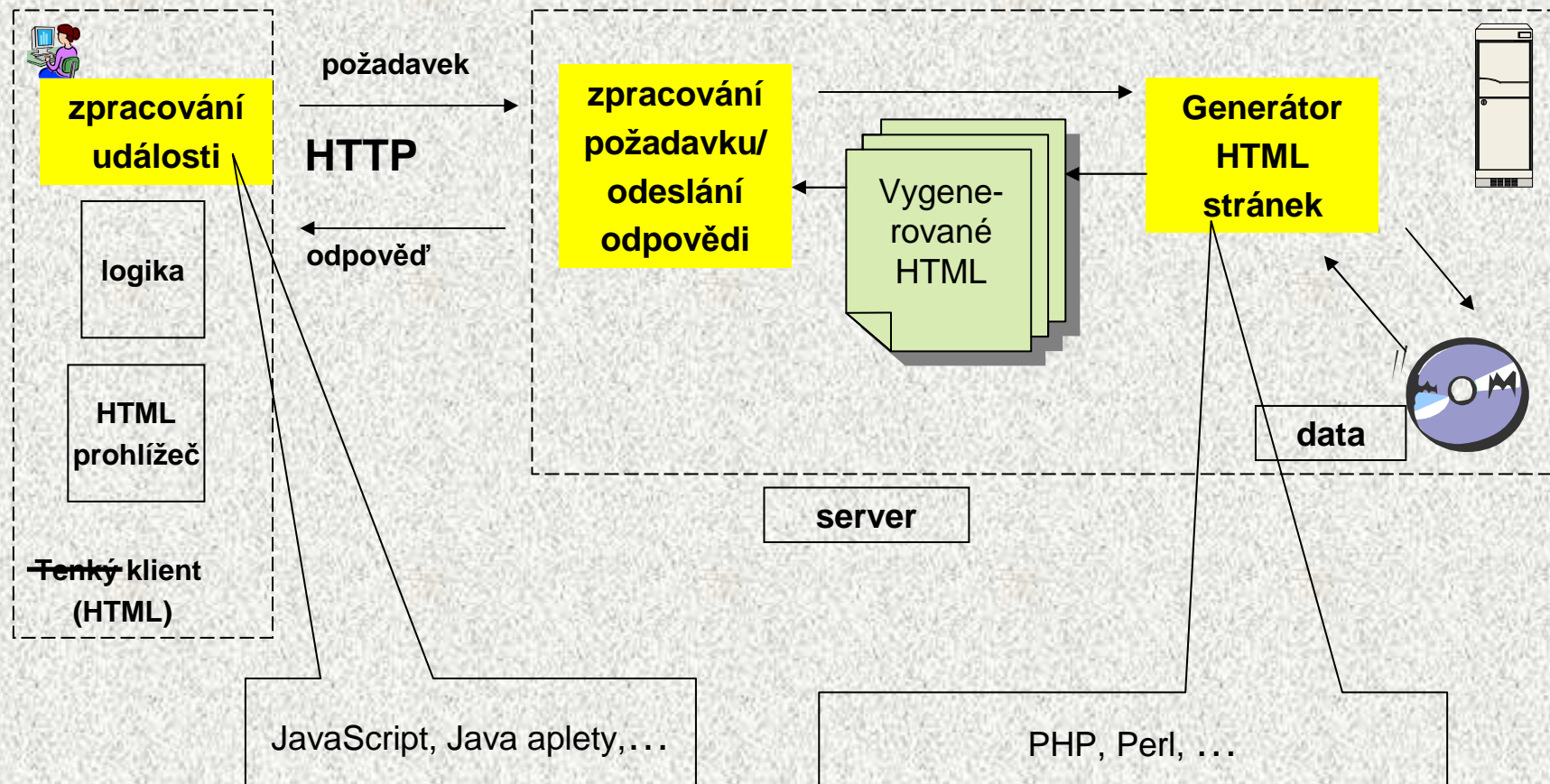
# Architektura web aplikace, statický web



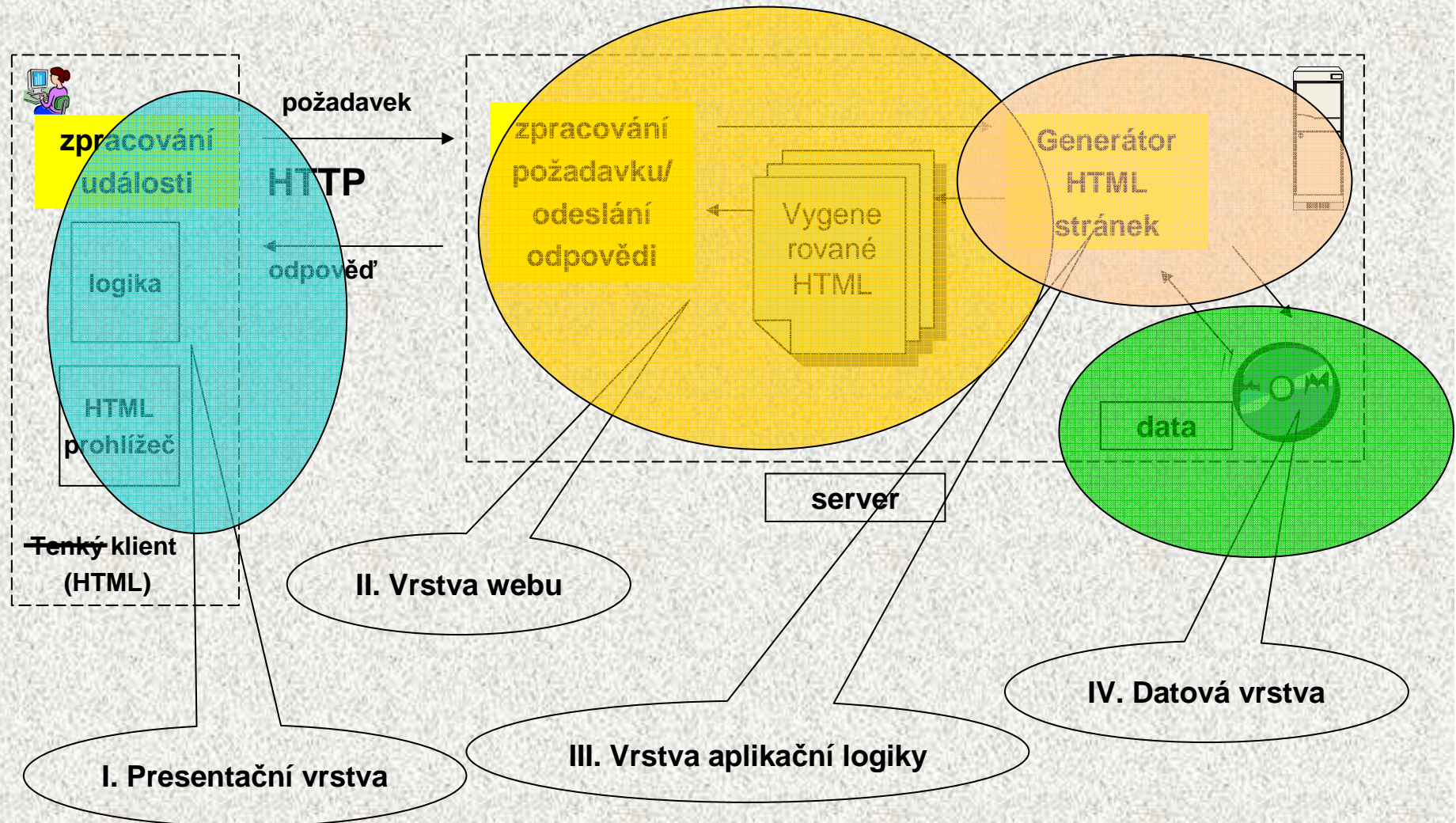
# Architektura web aplikace, dynamický web



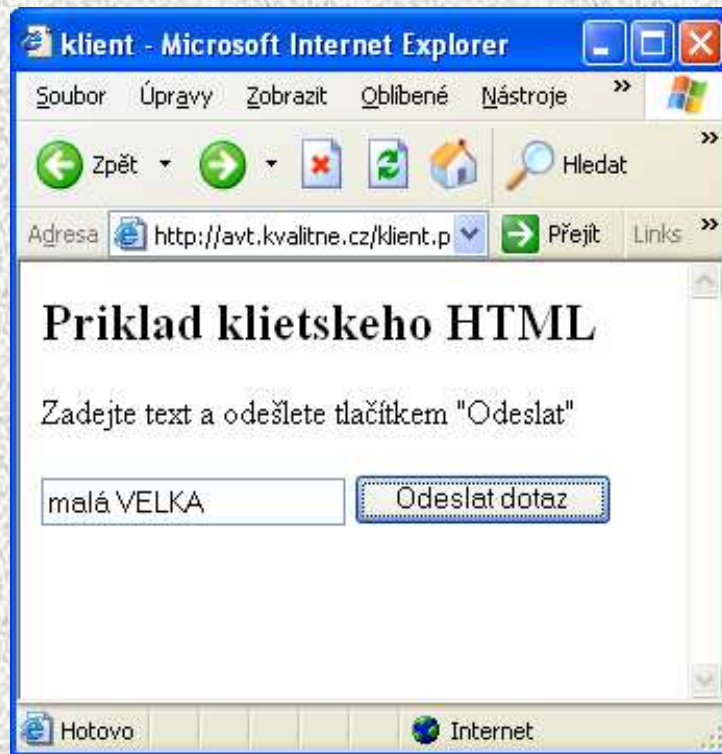
# Architektura web aplikace, dynamický web



# Architektura web aplikace, dynamický web



# Příklad klientské HTML stránky a reakce na ni





# Příklad klientské HTML stránky – klient.html

```
<html>
<head>
  <title> klient </title>
</head>
<body>
<h2> Příklad klientského HTML </h2>
Zadejte text a odešlete tlačítkem "Odeslat"
<form name="Zadani"  action="zobraz.php" method="post">
  <input type="text" name= "Text" value="" />
  <input type="submit" name= "Odeslat" />
</form>
</body>
</html>
```

# Příklad serverového programu - zobraz.php

```
<html>
<head>
  <title> server </title>
</head>
<body>
<h2> Příklad serverového PHP programu </h2>
Zadaný text byl: <?php echo $Text; ?><br />
Upraven na velká: <?php echo strtoupper($Text); ?>
<p><a href="klient.html"> Znovu zadávání</a></p>
</body>
</html>
```

# Vygenerovaná HTML stránka z zobraz.php

```
<html>
<head>
  <title> server </title>
</head>
<body>
<h2> Příklad serverového PHP programu </h2>
Zadaný text byl: malá VELKA<br />
Upraven na velká: MALá VELKA
<p><a href="klient.htm"> Znovu zadávání</a></p>
</body>
</html>
```

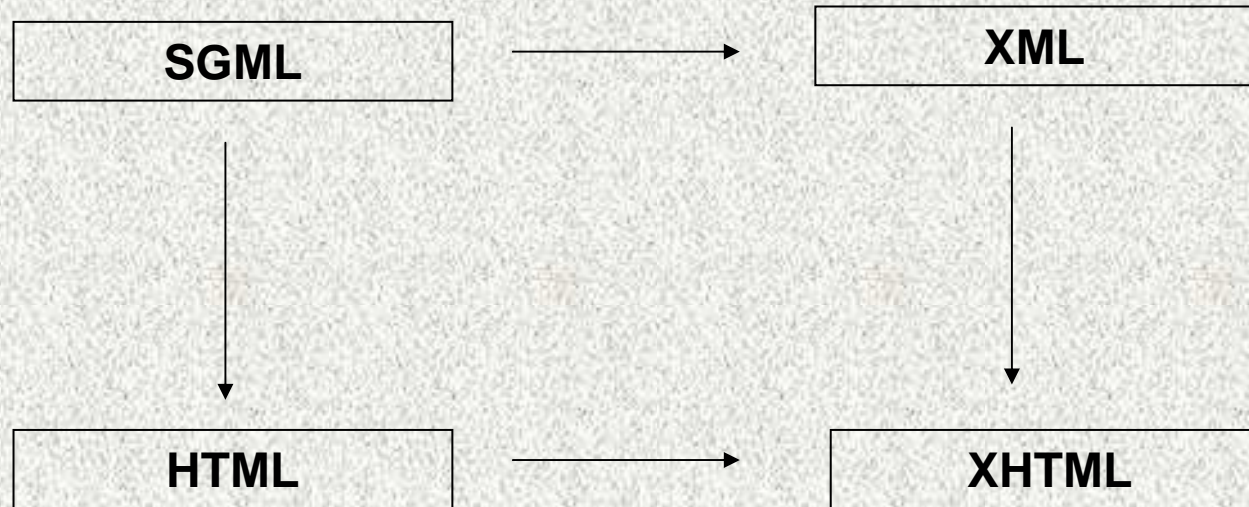
# Základní princip webové aplikace

- Funkce serveru
  - server zajistí aktivaci programu, resp. interpretaci programu požadovaného klientem (program nejčastěji v PHP)
    - převezme zaslané parametry resp. uloží je
    - spustí požadované služby
      - » dotazy do databází, komunikace s jinými servery
  - vygeneruje HTML stránku, do které uloží požadované údaje
    - i s případnými JavaScripty, Java Aplety
      - » tzv. dynamické HTML stránky
  - zašle stránku HTML klientovi k interpretaci – zobrazení
  - ukončí relaci
- Funkce klienta
  - zobrazí zaslanou HTML stránku
  - interpretuje skripty či aplety
  - zpracuje data zadávaná uživatelem
  - zašle zprávu zpět serveru (či jinému serveru – viz pojem portál)

# Jazyk HTML (HyperText Markup Language)

- Značkovací jazyk
- Vychází ze SGML (Standard Generalized Markup Language), je jeho aplikací
- poslední verze HTML 4.01
- Umožňuje u webového dokumentu:
  - snadný popis **struktury i obsahu** webového dokumentu
  - popis odkazů na další webové dokumenty
  - využití apletů, skriptů a jiných programů
- Umožňuje vytvářet zcela universální hypertextovou strukturu webových dokumentů

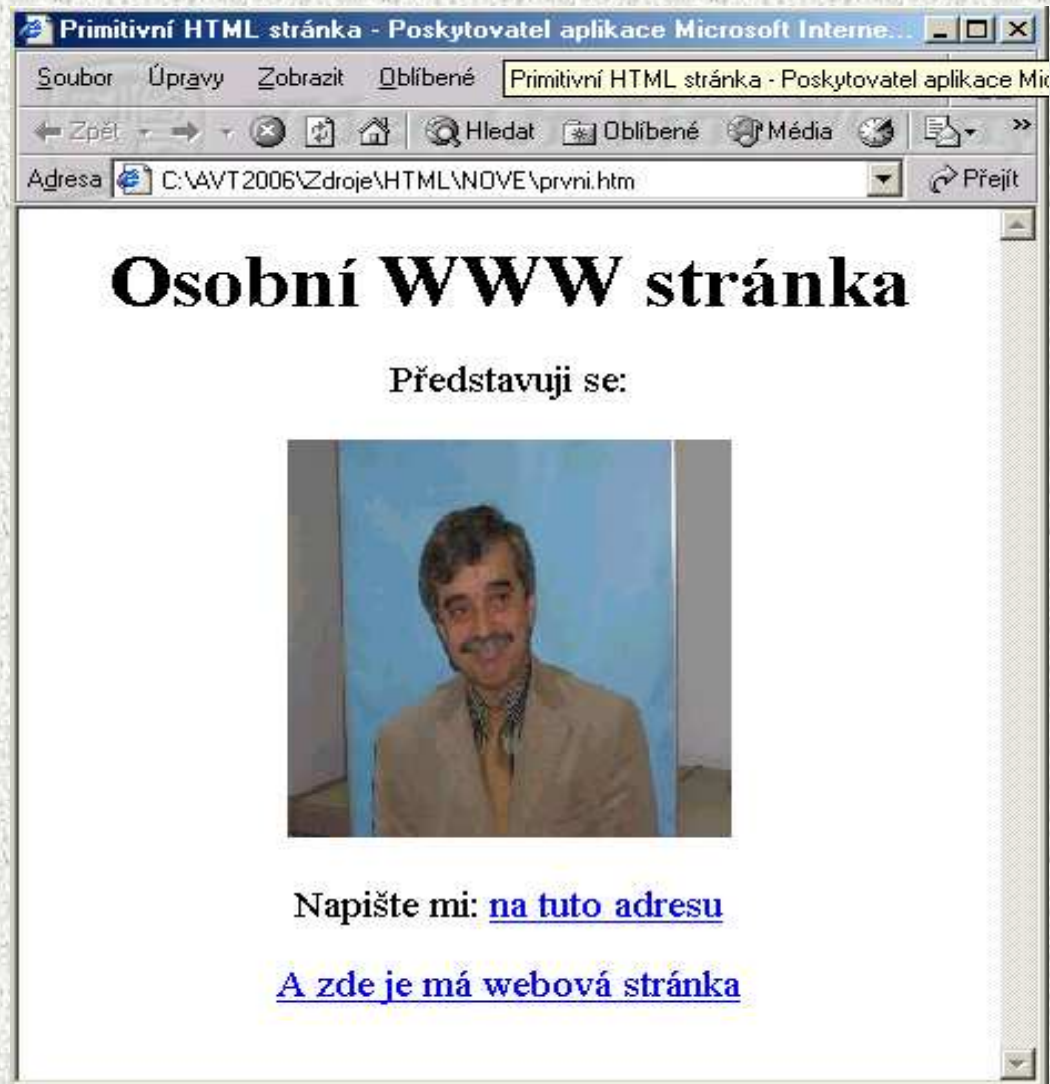
# HTML, SGML, XML, XHTML



# Primitivní příklad

```
<HTML>
<HEAD>
<TITLE>Primitivní HTML stránka</TITLE>
</HEAD>
<BODY>
<CENTER>
<H1>Osobní WWW stránka</H1>
<P>Představuji se:</P>
<IMG SRC="jelinek.jpg" width="200" height="200">
<P>Napište mi:
<A HREF="mailto:jelinek@cs.felk.cvut.cz">na tuto
                                adresu</A></P>
<P><A HREF="http://sgi.felk.cvut.cz/staff/jelinek.html">
                                A zde je má webová stránka </A></P>
</CENTER>
</BODY>
</HTML>
```

# Primitivní příklad – prvni.html





# Základní značky HTML

- **Dokument** `<html>`, `<head>`, `<body>`, `<title>`
- **Nadpisy** `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>`
- **Text** `<p>`, `<span>`, `<div>`, `<br>`
- **Obrázky** `<img>` (`<object>`)
- **Odkazy** `<a>`
- **Výčty** `<ul>`, `<ol>`, `<li>`, `<dl>`, `<dt>`, `<dd>`
- **Tabulky** `<table>`, `<tr>`, `<td>`, `<th>`
- **Formuláře** `<form>`, `<input>`, `<button>`, `<select>`, `<option>`,  
`<textarea>`

## Prostředky pro tvorbu HTML dokumentů

- textový editor
- editory specializované (např. PSPad)
- WYSIWYG (FrontPage, apod.)

# Základní skelet HTML

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>Základní skelet HTML stránky</TITLE>
```

```
</HEAD>
```

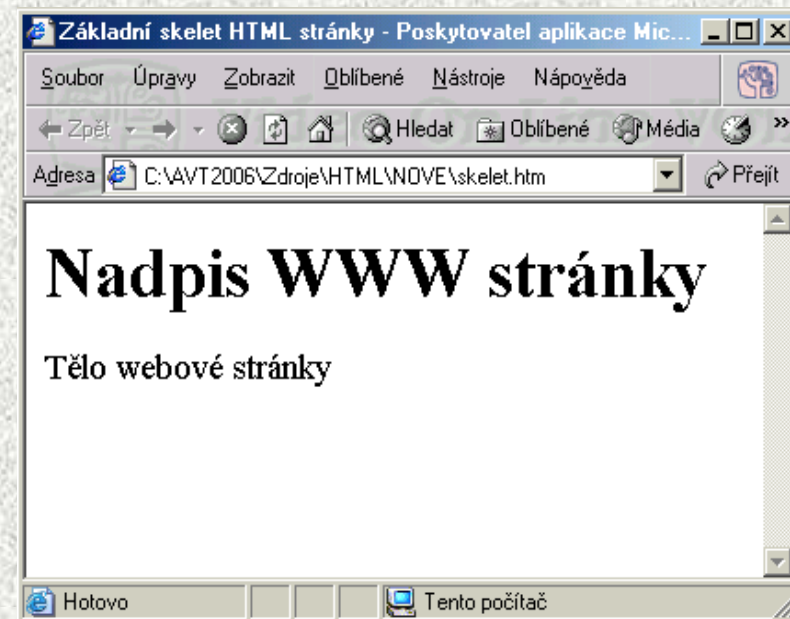
```
<BODY>
```

```
<H1>Nadpis WWW stránky</H1>
```

```
Tělo webové stránky
```

```
</BODY>
```

```
</HTML>
```



# Aplety

- Krátké aplikace včleněné do HTML kódu a spustitelné webovými prohlížeči
- Komunikace s aplety je založena na *zpracování událostí* a na *grafickém uživatelském rozhraní*
- Aplety mají některá omezení a „přednosti“ ( lze zmírnit a nastavit) :
  - **nepřístup** k systémovým proměnným, nelze `System.getProperty()`
  - **nepřístup** a **nezapisování** do souborů lokálního disku, kde běží aplet, ale má přístup k serverovým souborům
  - **nesmí** spouštět programy hostitele
  - **nenavazování** síťových spojení mimo domovský server
  - zobrazení apletů může trvat déle – řešení - komprimace do .jar
- Výhody:
  - nekladou žádné požadavky na instalaci, nezávislé na platformě
  - z principu nemůže ničit hostitelský systém
  - možnost elektronického podpisu
    - Zabezpečovací systém Javy  
(`java.security.AccessControlException`)

# Oblasti použití apletů

- Načítání datových souborů ze serveru daného URL
- Zobrazování dynamických stavových řetězců
- Vyhledání a spouštění jiných apletů
  
- Předzpracování dat před odesláním
- Jednodušší uživatelské aplikace
- Přehrávání videí, zvuků a zobrazování obrázků
- Získávání parametrů z HTML stránky
- Zjišťování stavu živorního cyklu apletu

# Nejjednodušší aplet pomocí AWT

```
import javax.swing.JApplet;  
import java.awt.*;
```

```
+--java.awt.Panel  
+--java.applet.Applet  
+--javax.swing.JApplet
```

```
public class Aplet1 extends JApplet {  
    // potomek třídy JApplet - Swing  
    public void paint(Graphics g){  
        g.drawString("Nejjednodušší aplet", 10, 20);  
    }  
}
```

- Metodě `paint(Graphics g)` je předán objekt `g` (grafický kontext), není volána uživatelem, ale prohlížečem, Překrytím této metody se vytvoří obsah okna

# Soubor HTML (odpovídající)

```
<HTML>
```

```
<BODY>
```

```
Text <b>před</b> apletem
```

```
<p>
```

```
<APPLET CODE="pr2_06/Aplet1.class" WIDTH=200 HEIGHT=30>
```

```
</APPLET>
```

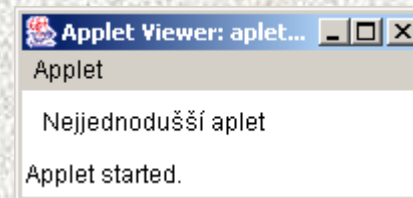
```
<p>
```

```
Text <b>za</b> apletem
```

```
</BODY>
```

```
</HTML>
```

A výsledek ...



# Nejjednodušší aplet (SWING)

```
public class Aplet2 extends JApplet {  
    JLabel jLabel1 = new JLabel();  
    public Aplet2 () {  
        jLabel1.setText("Jednoduchý aplet pomoci Swing");  
        this.getContentPane().add(jLabel1, BorderLayout.CENTER);  
    }  
}
```

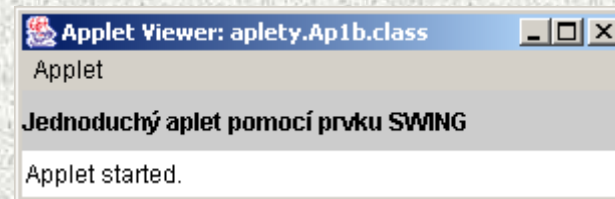
Kde je metoda paint?

Je implementována ve třídě JLabel

# Soubor HTML (odpovídající)

```
<HTML>
<BODY>
Text <b>před</b> apletem
<p>
<APPLET CODE="Aplet2.class" WIDTH=300 HEIGHT=30>
</APPLET>
<p>
Text <b>za</b> apletem
</BODY>
</HTML>
```

A výsledek ...



Výhody použití JLabelu:

- možnost vkládat složité výrazy včetně HTML
- možnosti využití dalších vlastností (zarovnání textu)



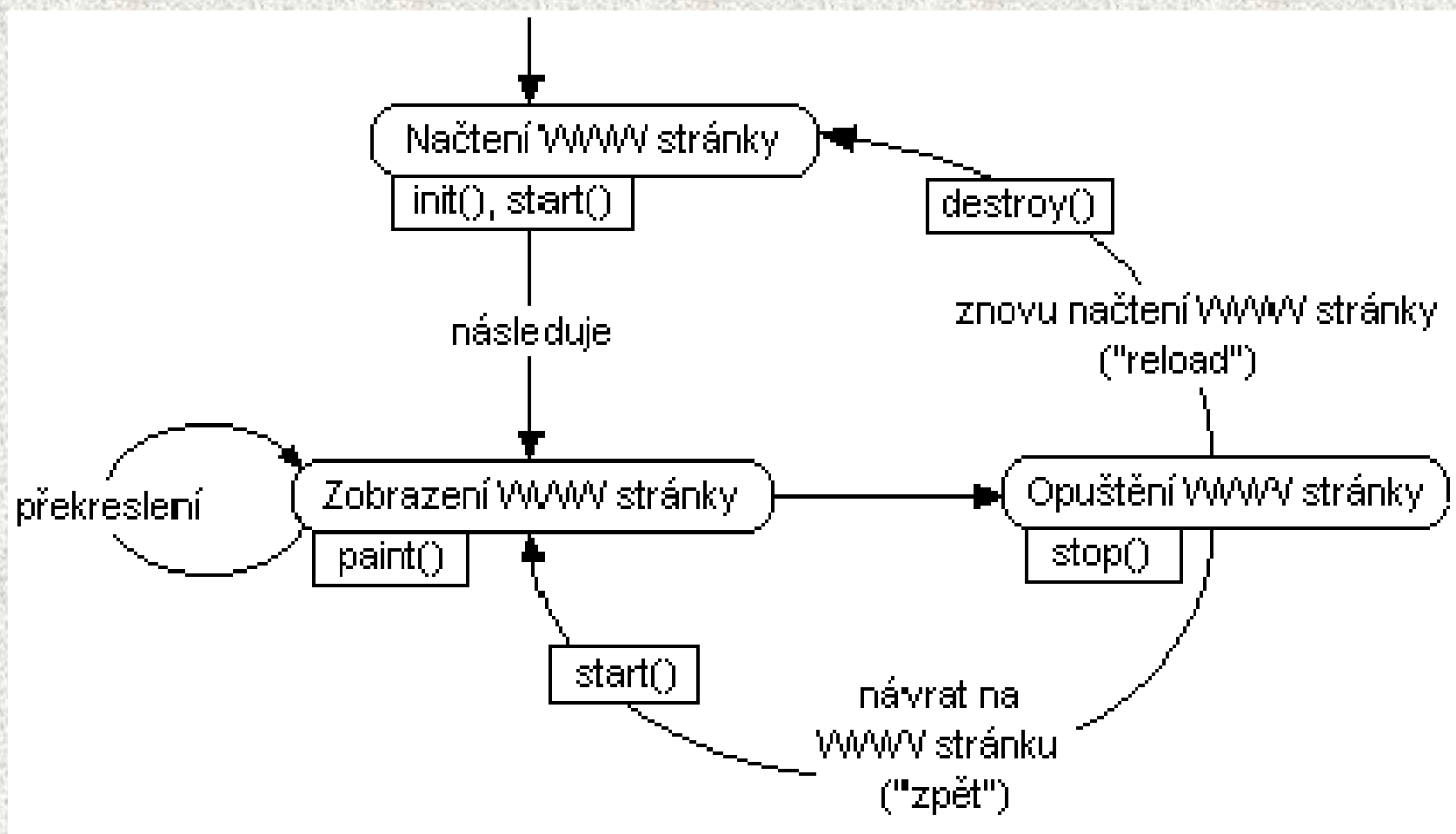
# Vlastnosti apletů

- Aplety (Swing) jsou potomky třídy JApplet a ta je potomkem třídy Applet (AWT)
- Třída Applet je potomkem třídy Panel – viz Grafické uživatelské rozhraní
- Aplety importují třídy java.awt a java.swing, pracují s kontejnery a komponentami - >> proto metoda paint s parametrem Graphics
- Aplet nemá metodu main
- Aplet je určen ke spuštění na webové stránce
- Aplet je spuštěn nejčastěji tak, že je vyvolána metoda paint pro překreslení okna třídy uvedené v:

```
<APPLET CODE="applet.Ap1.class" WIDTH=200 HEIGHT=30 </APPLET>
```

- V každém HTML souboru může být libovolný počet apletů
- Aplet můžeme zobrazit i tzv. **appletviewery**, aplikacemi pro ladění

# Životní cyklus apletu



# Standardní „minimální“ struktura apletu

```
public class MinimálníAplet extends Applet {
    public void init() {
        // voláno při prvním použití apletu, inicializace proměnných
    }
    public void start() {
        // voláno po init při každé nové návštěvě příslušné webové stránky
        //(po znovuootevření, po překrytí), spuštění procesů, ...
    }
    public void paint(Graphics g) {
        // vlastní vykreslování, volána prohlížečem automaticky,
        // je-li třeba překreslit stránku, možno spustit pomocí repaint()
    }
    public void stop() {
        // použito po zmizení webové stránky s tímto apletem, procesy běží
        // dál, aplet se opět spouští pomocí start()
    }
    public void destroy() {
        // voláno při odstranění apletu z paměti, zrušení činnosti vláken
    }
}
```

# Životní cyklus apletu

- Aplety mívají různou strukturu danou programátorem a příslušným vývojovým prostředím
- **Aplet vyvolává prohlížeč automaticky v závislosti na stavu webové stránky**
- Nejjednodušší aplet musí mít část pro zobrazení nějaké informace např.pomocí `paint()`

```
public void init() {
```

- metoda použita při inicializaci apletu, je volána po konstruktoru (je-li nějaký), načte ze stránky HTML parametry, inicializace proměnných a vykreslení komponent, apod.

```
public void start() {
```

- spouští činnost apletu, voláno po `init()`, volána po každém "oživení" webové stránky (po ikonizaci, posun stránky, spouštění vláken s animací apod.

```
public void paint(Graphics g) {
```

- vlastní vykreslování, spolupracuje s metodami `repaint()` a `update()`, lze vyvolat i přes `repaint()`

```
public void stop() {
```

- zastavení činnosti při dočasném „umrtvení“ příslušné stránky, po ikonizaci, posunu stránky, ukončení činnosti v komunikaci s pamětí, vlákna mohou běžet dále

```
public void destroy() {
```

- zrušení apletu, uvolnění paměti, vlákna podle aplikace

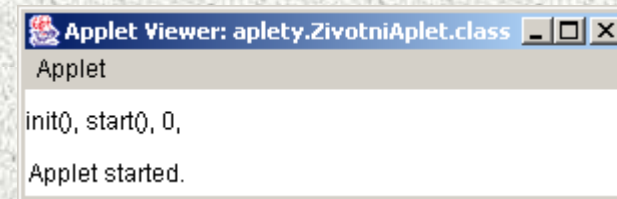
# Životní cyklus apletu – příklad I

```
public class ZivotniAplet extends JApplet {
    String stav;
    int pocet_paint;

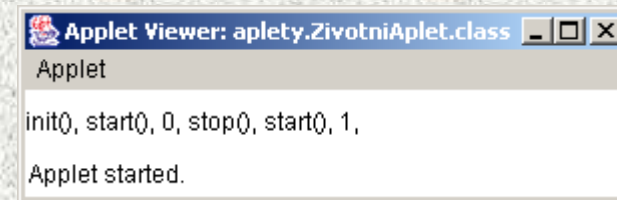
    public void init() {
        stav = new String();
        pocet_paint = 0;
        stav = stav.concat("init(), ");
        // zřetězení, stav+="init(), "
        System.out.println(stav);
        repaint();
    }
    public void start() {
        stav = stav.concat("start(), ");
        System.out.println(stav);
        repaint();
    }
}
```

# Životní cyklus apletu – příklad II

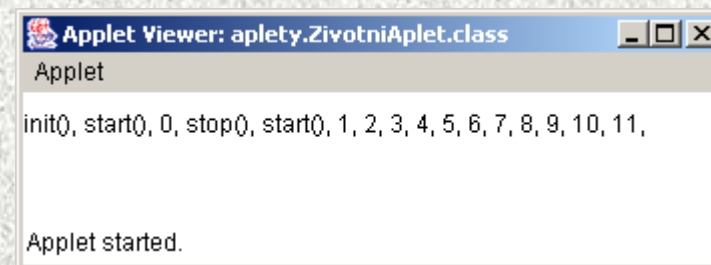
```
public void paint(Graphics g) {
    stav+=(pocet_paint++) + ", ";
    g.drawString(stav, 0, 20);
    System.out.println(stav);
}
public void stop() {
    stav += "stop(), ";
    System.out.println(stav);
    repaint();
}
public void destroy() {
    stav += "destroy(), ";
    System.out.println(stav);
    repaint();
}}
```



minimalizace



roztažení



## Činnost prohlížeče při zpracování `<applet>`

- Vyhradí místo na displeji o určené šířce a výšce
- Přenese bytecode specifikovaného apletu (.class)
- Vytvoří instanci této třídy
- Zavolá instanční metody `init()` a `start()`, nejsou-li pak rovnou `paint()`
- Aplet vyvolán prohlížečem automaticky v závislosti na stavu webové stránky

# Parametry tagu applet v HTML souboru

Minimální parametry:

- **CODE** - jméno třídy, která má být spuštěna
- **WIDTH, HEIGHT** - šířka a výška okna apletu

Další parametry:

- **CODEBASE** - URL, kde se nachází soubory tříd apletu
- **ALT** - alternativní text pro případ, že prohlížeč není schopen zobrazit aplet
- **ALIGN** - relativní zarovnání apletu na webové stránce
  - (left, right, top, absmiddle, bottom)
- **HSPACE, VSPACE** - odsazení v pixelech
- **ARCHIVE** – umístění komprimovaného archívu .jar
- atd.



# Předávání parametrů apletu

- Parametry se předávají v mezi značkami `<APPLET>` a `</APPLET>`
- Parametr **NAME** v kombinaci s hodnotou pomocí **VALUE**

```
<APPLET CODE="MuJApplet.class" WIDTH=120 HEIGHT=120
  <PARAM NAME="obr1" VALUE="obrazek1.png">
  <PARAM NAME="obr2" VALUE="obrazek2.png">
</APPLET>
```

- V apletu pak voláme soubory **obrazek1** a **obrazek2** v prvním volání přes
  - `String jmenoObr1 = getParameter("obr1");`
  - `String jmenoObr2 = getParameter("obr2");`
- Volání souborů z jiného než adresáře, kde se nachází aplet se děje přes **CODEBASE**:
  - `Image obr = getImage(getCodeBase(), "obrazek.jpg");`
- Na stavovém řádku můžeme prezentovat informace pomocí
  - `showStatus (String stav);`

Například: `showStatus ("Probíhá kopírování ....")`

# Příklad použití parametrů v HTML I

```
<HTML>
<BODY>
<APPLET CODE="Aplety/ApletKocka.class" WIDTH=80
    HEIGHT=100
        CODEBASE="classes"
        ALIGN="right"
        HSPACE=20
        VSPACE=30
    >
<PARAM NAME="obrazek" VALUE="kocka.jpg">
</APPLET>
</BODY>
</HTML>
```

# Příklad použití parametrů v HTML II

```
public class ApletKocka extends JApplet {  
    String jmObr = null;  
    Image img = null;  
    public void init() {  
        jmObr = getParameter("obrazek");  
        img = getImage(getCodeBase(), jmObr);  
    }  
    public void paint(Graphics g) {  
        Dimension d = getSize(); // vrací rozměr z HTML  
        if (img != null)  
            g.drawImage(img, 0, 0, d.width, d.height, this);  
        showStatus(jmObr); // dej jméno obrázku  
    }  
}
```



# Příklad použití parametrů v HTML III



# Příklad použití výsledku z apletu v HTML

```
<html>
  <head>
    <title>Applet HTML Page</title>

    <script type="text/javascript">

function nactiDataZApletu(){
  document.formular.otazky.value = document.mujAplet.getCelkovyPocetOtazek();

document.formular.odpovedi.value =
                                document.mujAplet.getCelkovyPocetSpravnychOdpovedi();

  document.formular.hodnoceni.value = document.mujAplet.getHodnoceni();

  return true;
}
    </script>

  </head>
  <body>
```

# Příklad použití výsledku z apletu v HTML

```
<h3>Applet HTML Page</h3>
<p><applet name="mujAplet" codebase="." code="pr2_06/NewJApplet.class"
width=350 height=200></applet> </p>

<form name="formular" action="">
  <table border="1">
    <tr> <td><label for="otazky">Otázky</label></td>
      <td><input type="text" value="" name="otazky" /></td>
    </tr>

    <tr><td><label for="odpovedi">Odpovědi</label></td>
      <td><input type="text" value="" name="odpovedi" /></td>
    </tr>

    <tr> <td><label for="hodnoceni">Hodnocení</label></td>
      <td><input type="text" value="" name="hodnoceni" /></td>
    </tr>
  </table>

  <input type="button" onclick="nactiDataZApletu()" value="Načíst data z apletu"
/>
</form>
</body>
</html>
```

Pro zájemce

# Příklad použití výsledku z apletu v HTML

```
public class NewJApplet extends JApplet {
String hodnoceni;
String celkovyPocetOtazek;
String celkovyPocetSpravnychOdpovedi;
@Override
public void init() {
    double doubleHodnoceni;
    double doublePocetOtazek;
    double doubleSpravneOdpovedi;
celkovyPocetOtazek =
    JOptionPane.showInputDialog("Zadejte celkovy pocet otazek v testu");
celkovyPocetSpravnychOdpovedi =
    JOptionPane.showInputDialog("Zadejte celkovy pocet spravnych
    odpovedi");
doublePocetOtazek = Double.parseDouble(celkovyPocetOtazek);
doubleSpravneOdpovedi =
    Double.parseDouble(celkovyPocetSpravnychOdpovedi);
doubleHodnoceni = (doubleSpravneOdpovedi / doublePocetOtazek) * 100;
hodnoceni = Double.toString(doubleHodnoceni);
JOptionPane.showMessageDialog(null, "Vase hodnoceni " + hodnoceni +
"%", "Hodnoceni", JOptionPane.PLAIN_MESSAGE);
}
```

Pro zájemce

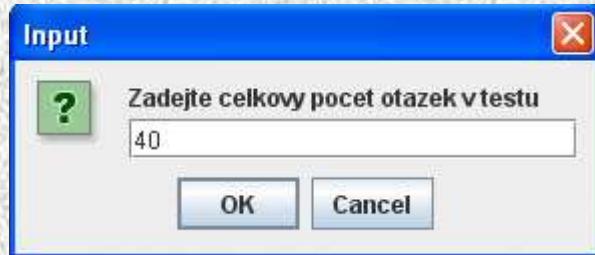
# Příklad použití výsledku z apletu v HTML

@Override

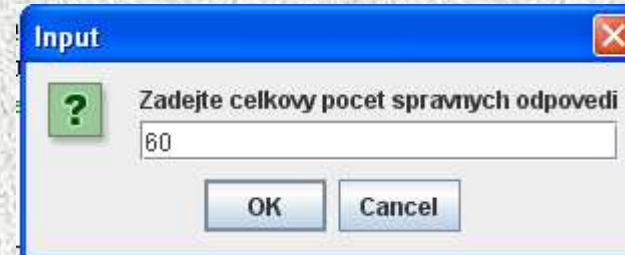
```
public void paint(Graphics g) {  
}  
  
public String getHodnoceni() {  
    return hodnoceni;  
}  
  
public String getCelkovyPocetOtazek() {  
    return celkovyPocetOtazek;  
}  
  
public String getCelkovyPocetSpravnychOdpovedi() {  
    return celkovyPocetSpravnychOdpovedi;  
}  
}
```



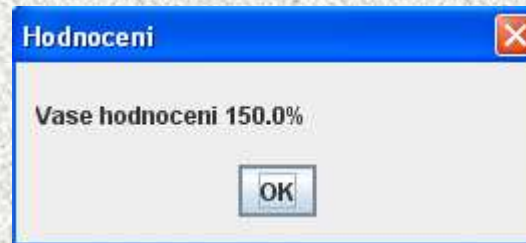
# Příklad použití výsledku z apletu v HTML



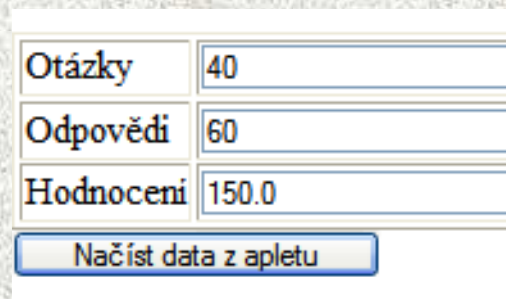
Input dialog box with title "Input". The text reads "Zadejte celkový počet otázek v testu". The input field contains the number "40". There are "OK" and "Cancel" buttons at the bottom.



Input dialog box with title "Input". The text reads "Zadejte celkový počet správných odpovědí". The input field contains the number "60". There are "OK" and "Cancel" buttons at the bottom.



Hodnocení dialog box with title "Hodnocení". The text reads "Vase hodnoceni 150.0%". There is an "OK" button at the bottom.



Otázky	40
Odpovědi	60
Hodnocení	150.0

Načíst data z apletu

# Spuštění apletu jako aplikace I

- Aplikaci v Javě lze použít jako aplet a naopak
- Aplikace v Javě musí mít `main` a naopak `main` nevadí apletu

```
<HTML>
```

```
<BODY>
```

```
Aplet a aplikace dohromady
```

```
<APPLET CODE="ApletAAplikace.class" WIDTH=100 HEIGHT=60>
```

```
</APPLET>
```

```
</BODY>
```

```
</HTML>
```

# Spuštění apletu jako aplikace II

```
public class ApletAAplikace1 extends JApplet implements
    ActionListener { // potomek třídy JApplet
    JButton jButZiju = new JButton("Žiju"); // definice tlačítka
    JLabel jLabKliku = new JLabel("Počet kliku: 0");
    // definice tlačítka

    static int pocet = 0;

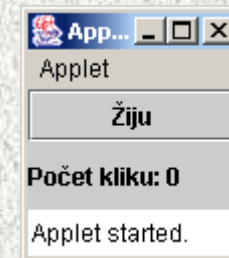
    public void init() { // inicializace apletu
        Container kon=getContentPane();
        kon.add(jButZiju, BorderLayout.NORTH); // tlačítko do okna
        kon.add(jLabKliku, BorderLayout.CENTER); // popisek do okna
        jButZiju.addActionListener(this);
        // registrace posluchače, předání odkazu na mne
    }
```

# Spuštění apletu jako aplikace III

```
public static void main(String[] args) {  
ApletAAplikace1 applet = new ApletAAplikace1();  
                                // vytvoření apletu  
JFrame window = new JFrame("AAA");  
window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
                                // zavření okna  
  
window.setSize(90,90);  
window.setContentPane(applet);  
applet.init();  
window.setVisible(true);  
}
```

```
public void actionPerformed(ActionEvent e) {  
jLabKliku.setText("Poč.kliku:" + ++AplatAAplikace1.pocet);  
}
```

výsledek



# Vložení apletu do HTML I

- **Pomocí značky (tagu) `<APPLET>` `</APPLET>`**
  - specializovaný pouze na aplety
  - od HTML 4.0 označen jako zastaralý (deprecated)
  
- **Pomocí značky `<OBJECT>` `</OBJECT>`**
  - univerzální, nejen na aplety
  - konverze html souboru s `<APPLET>` pomocí `htmlconverter`, součást JDK  
**(`JDK/lib/htmlconverter.jar`)**

# Vložení apletu do HTML II

- Starší způsob

```
<APPLET code="aplety.ApletKocka.class"
        width=300 height=200>
<PARAM NAME="obrazek" VALUE="kocka.jpg">
</APPLET>
```

- Nový způsob

```
<OBJECT codetype="application/java"
        CODE="aplety.ApletKocka.class"
        width="300" height="200">
<PARAM NAME="obrazek" VALUE="kocka.jpg">
</OBJECT>
```

# Zmírnění omezení apletů

- Vytvořením souboru `.java.policy` v domovském adresáři uživatele:
  - Tento textový soubor obsahuje položky ve tvaru:

```
grant codeBase "http://127.0.0.1/-" {  
    permission java.security.AllPermission;  
};
```
  - nastavuje oprávnění pro jednotlivé servery
- Podepsání apletu digitálním certifikátem:
  - `keytool -genkey -alias nazev_klice // vygenerování vlastního klíče`
  - `jarsigner soubor.jar nazev_klice`  
**// podepsání celého archívu, samotnou třídu nelze**
  - po načtení apletu je uživatel požádán o přijetí certifikátu
- Spuštění apletu jako aplikace:
  - -aplikace má stejná práva jako uživatel