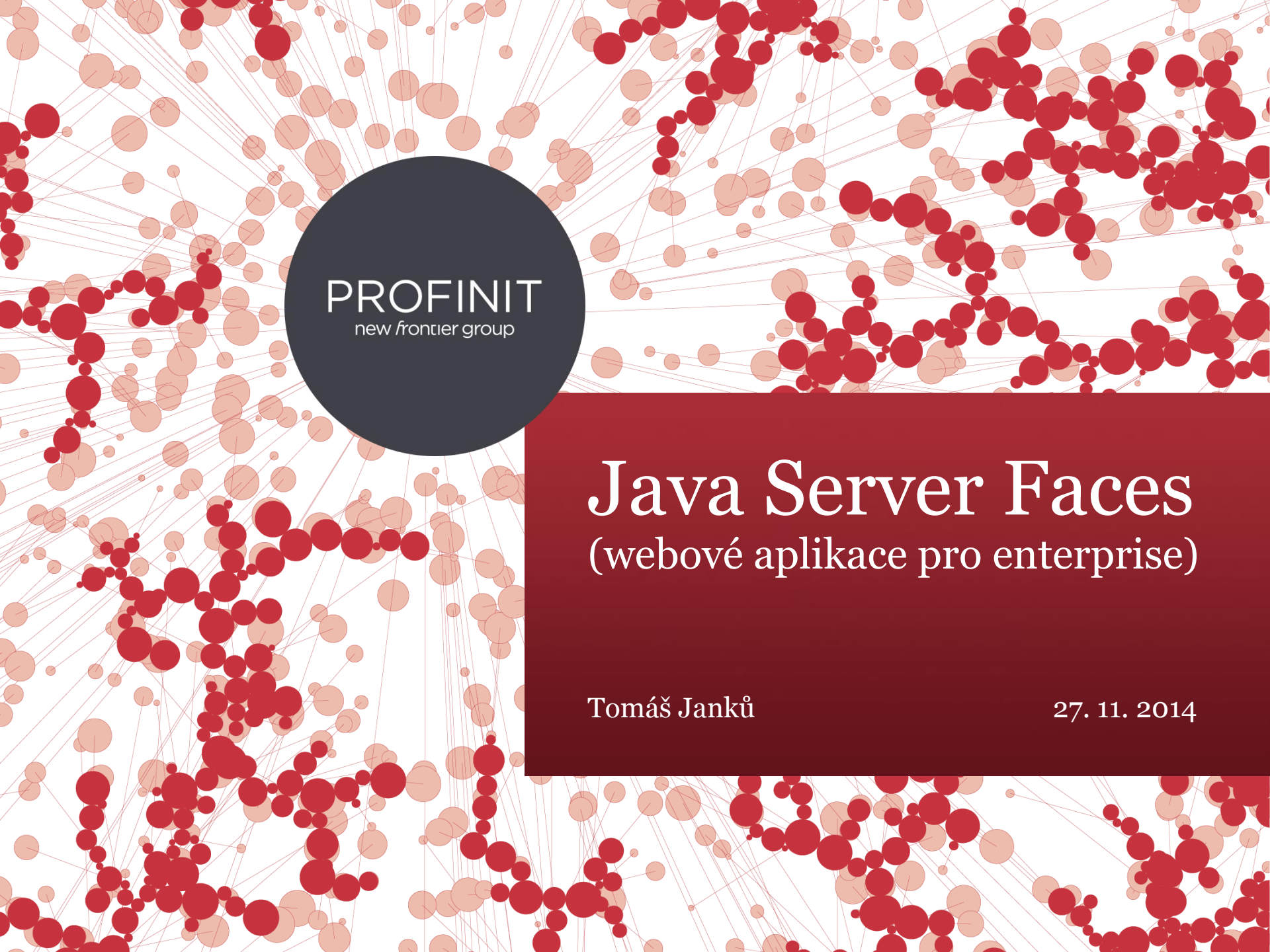


Java Server Faces
(webové aplikace pro enterprise)

Tomáš Janků

27. 11. 2014

Kdo jsem?

PROFINIT
new frontier group

new frontier group

PROFINIT
new frontier group

IT konzultant
(4+ let)

- Java
- Spring
- Oracle
- Angular JS

0

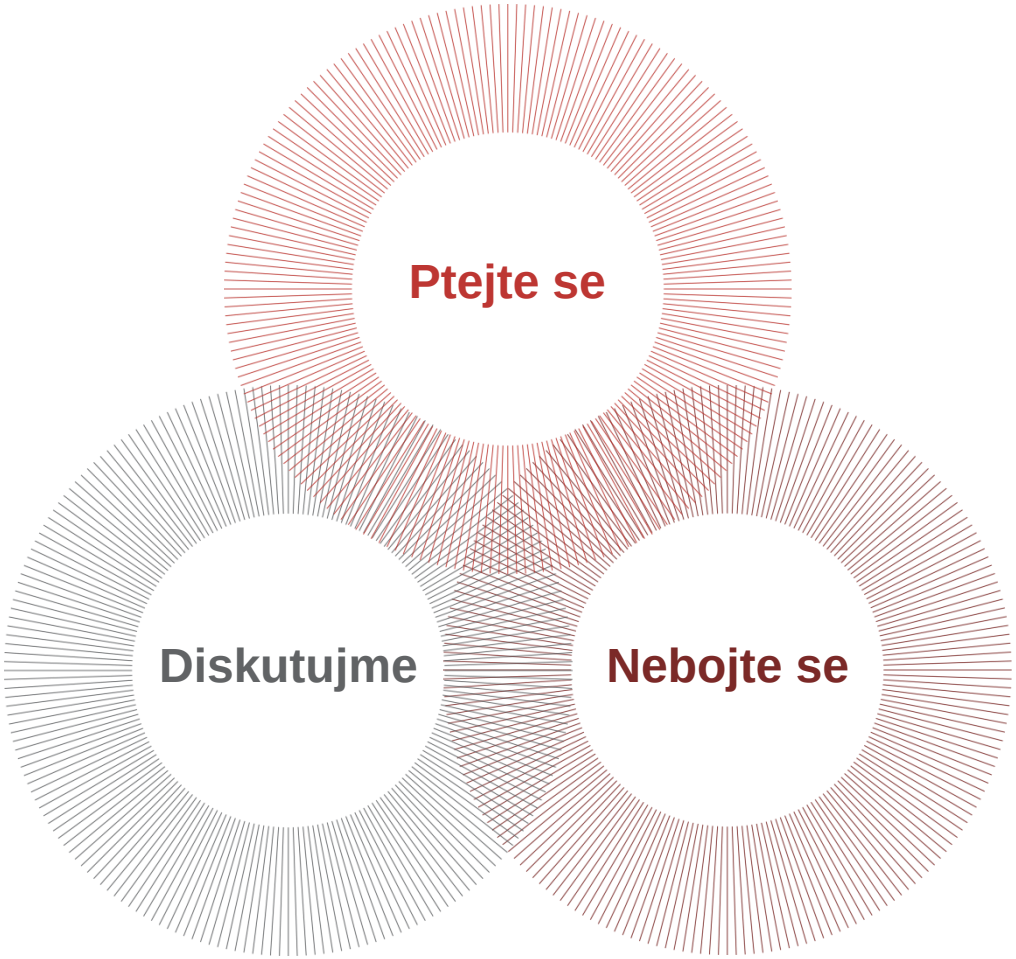
Co nás v příštích 2,5 hod. čeká ?

Agenda přednášky

PROFINIT
new frontier group

- Obecné povídání o webových aplikacích v „enterprise“ prostředí
- Aktuální trendy ve vývoji webových aplikací
- Frameworky pro vývoj frontendu
- Java Server Faces (frontendový framework standardu Java)
  - Proč, kdy a k čemu jej použít
  - Architektura a koncepty frameworku
  - Co nám nabízí a jak začít
  - Facelety, beany a jiná „sprostá“ slovíčka
- Integrace s jinými frameworky
- Cokoliv Vás bude ještě zajímat a bude na to čas...

Prosba: Buďte aktivní…



1 Enterprise prostředí

Co je to „enterprise“ ?

PROFINIT
new frontier group

- Enterprise ⇔ Organizace
- Tvorba aplikací pro potřeby organizace, ne individuálních uživatelů
- Co chce koncový uživatel?
  - Moderní vzhled
  - Pohodlnost práce
  - Rychlost
- Co chce organizace?
  - Stabilitu a bezchybnost
  - Bezpečnost
  - Spolehlivost
  - Škálovatelnost a adaptabilita
  - Integrovatelnost

Specifika „enteprise“ prostředí

- Stabilita
  - Definované SLA (“service level agreement”)
  - Postihy za porušení
  - Např. 99,9% ⇔ 8,76 hod.
- Bezchybnost
  - Vícekolové testování (FAT, xUAT)
  - Akceptační kritéria
  - Podpora produkce
- Bezpečnost
  - Citlivost dat (legislativa, osobní údaje, byznys data)
  - Šifrování
  - Spravovatelná a težce podvrhnutelná autentizace/autorizace
  - Auditovatelnost
  - Standardy (např. naplnění ISO)

Specifika „enteprise“ prostředí

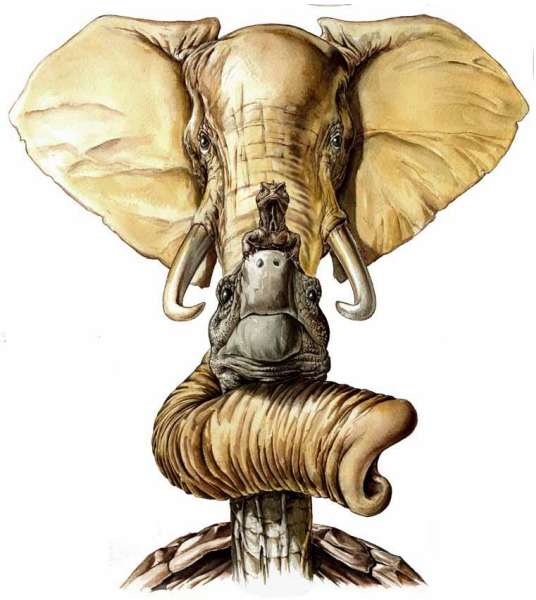
PROFINIT
new frontier group

- Spolehlivost
  - V souladu s procesy
  - Typicky úspora ⇔ nahrazení lidských zdrojů aplikací (úspora FT)
- Škálovatelnost a adaptabilita
  - Nutnost plynule reagovat na změny v procesech
  - Rozšiřitelnost bez narušení funkčnosti
  - Kompozice dílčích komponent/služeb
  - Procesní schémata
  - Zvládat navyšující se nároky
  - Zvládat změny technologií
- Integrovatelnost
  - Schopnost spolupracovat s ostatními systémy
  - Využívání existujících procesů
  - Napojení na řešení třetích stran

Životnost “enterprise” aplikace

PROFINIT
new frontier group

- Dlouhověké aplikace
  - Hlavní vývoj
  - Následně jen údržba a drobný rozvoj (v řádu let)
  - Postupně se stabilizuje
- Změna ⇔ Riziko
  - O peníze jde až v první řadě
  - Proč měnit, co funguje?
    - Nutno přesvědčit “byznys” (aby nám to zaplatil)
    - Nutno mít byznys case ⇔ vyjádření přínosů v penězích
- Postupně rostoucí problémy
  - Zastarává technologie
    - Enterprise je pár let za trendy
  - Mizí a zdražuje pracovní síla
    - Odchod zaměstnanců
    - Nová pracovní síla nepřibývá



Typy „enterprise“ zákazníků

- Velcí zákazníci (>> 1 000 000,- per projekt)
  - Státní správa
  - Banky
  - Pojišťovny
  - Česká pošta
- Středně velcí zákazníci (> 100 000,- per projekt)
  - Automobilky
  - Leasingové společnosti
  - Finanční poradenství
  - Logistika (letiště, nádraží, překladiště)
  - Zdravotnické organizace (nemocnice, lékárny)
- Malí zákazníci (> 10 000,- per projekt)
  - OSVČ, s.r.o, apod.

Jaké jsou v tom peníze, aneb proč Java?

PROFINIT
new frontier group

- Mzda srovnání (ČSU 2013) – průměr ČR

Pozice	Nástupní	Po 3+ letech
PHP programátor	24 544 Kč/měs.	31 177 Kč/měs.
Java/.NET programátor	25 589 Kč/měs.	38 204 Kč/měs.
C/C++ programátor	31 151 Kč/měs.	37 137 Kč/měs.
IT analytik	27 967 Kč/měs.	38 871 Kč/měs.
DB analytik	26 552 Kč/měs.	36 146 Kč/měs.
Projektový manažer	29 789 Kč/měs.	49 359 Kč/měs.

- Ocenění projektu => MD (man-day rate)
  - 1MD v bankovnictví v ČR ⇔ 8 000,- - 12 000,-
  - Odhad ceny, nákladů na zdroje, času pro vypracování
  - Univerzální pojem srozumitelný všem

Enterprise projekty – realizace

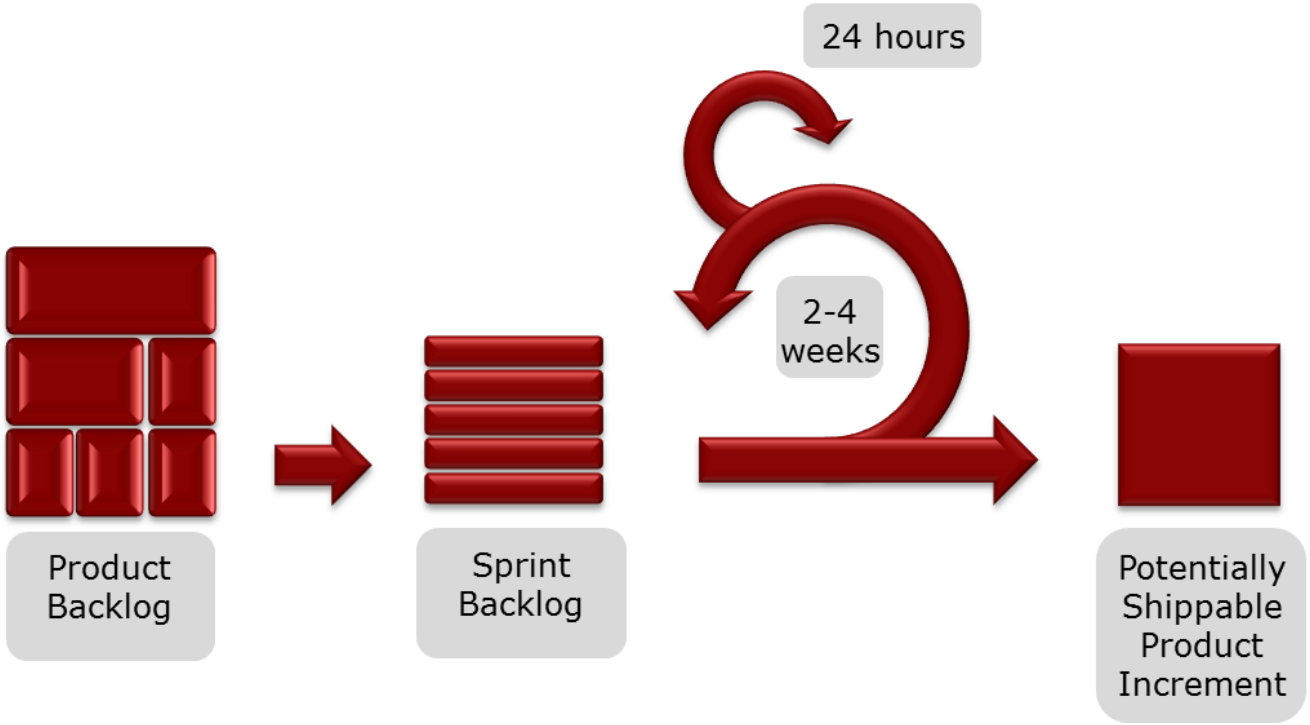
PROFINIT
new frontier group

- Interní vývoj
  - Často malá motivace
  - Omezený pracovní pool (zkušenosti, technologie)
  - Spíše jen údržba a menší rozvoj
- Bodyshop projekty
  - Interně projektový manažer
  - Externě najímáni technici dle CV/pohovoru
    - Možnost zvolit „člověka na míru“
    - Placen nájem dle manday-rate
  - Projekt plně v kompetenci projekt manažera
- Fixed-time, fixed-price
  - Interně jen zadání, externě vše ostatní
  - Vše plně v kompetenci dodavatele
  - Realizace na základě vstupů od zákazníka
  - Zákazník nemá moc možností projekt ovlivnit

Enterprise projekty – metodika SWI

PROFINIT
new frontier group

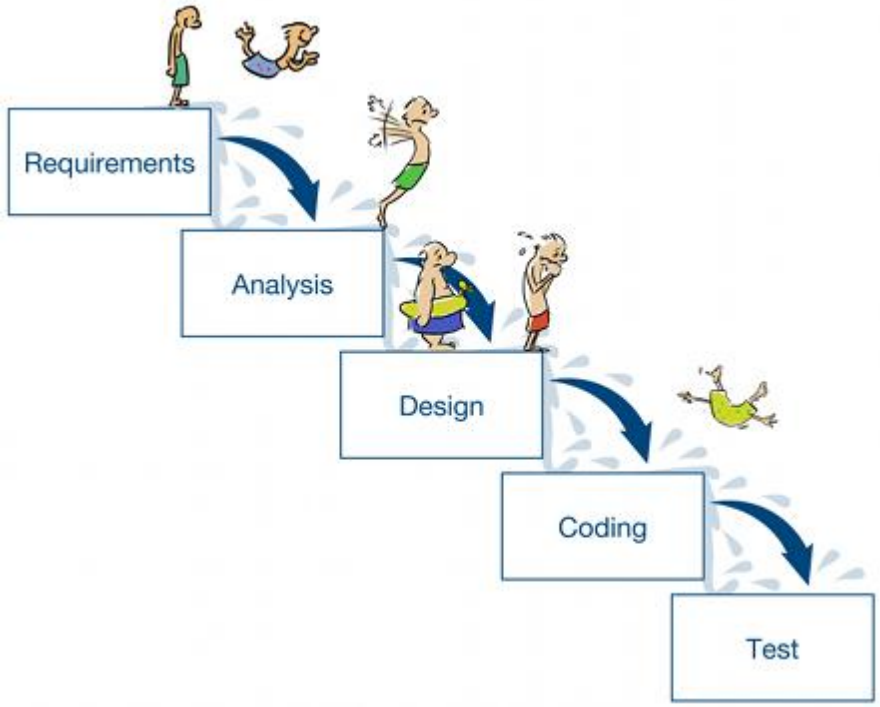
- Agilní vývoj
  - Spíše jen pro interní projekty
    - Máme placený zaměstnanecký pool
    - Postupně realizace příchozích požadavků
    - Možnost měnit v průběhu priority
  - Scrum



Enterprise projekty – metodika SWI

PROFINIT
new frontier group

- Waterfall
  - Spíše pro FTFP
    - Jsme vázáni poptávkou
    - Pevný budget
    - Pevný termín
  - FTFP projekty jsou ziskovější
    - Realizuje se „dohoda“
    - Změny = change request



- Smart FTFP (Profinit)
  - Waterfall se silným zapojením zákazníka
    - Lepší vztah se zákazníkem
    - Minimalizace rizika, že nenaplníme potřeby zákazníka
  - Využití agilních metodik
    - Ale s pevným omezením ceny/termínů

Enterprise projekty – rizika

- 1 projekt = více dodavatelů (i interních) = 1 budget
  - Nutno s tím počítat
  - Nutno plánovat rizika
    - Součinnost
    - Neznalost prostředí/schopností ostatních dodavatelů
- Odhad ceny s pomocí „křišťálové koule“
  - Málo času prostudovat nabídku
  - Neznalost prostředí
  - Nesmíte podhodnotit, ani nadhodnotit
    - Často nutno odůvodnit
    - Dobré vyjít ze zkušeností z minulých projektů
    - Snížení ceny snížením rizik

Obecné rady do života…

PROFINIT
new frontier group

- Use the best tool available
- Keep it simple (KISS, Occamova britva)
  - Vytvářet co nejjednodužší bloky a ty kombinovat
- Dont repeat yourself (DRY)
- You aren’t gonna need it (YAGNI)
- Fail-fast and hard (FFAH)
  - Nepolykejte výjimky, nikdy, za žádných okolností
- Test, test, test
  - Nejlépe automaticky, nejlépe atomicky, nejlépe pořád
- Předčasná optimalizace je kořenem všeho zla

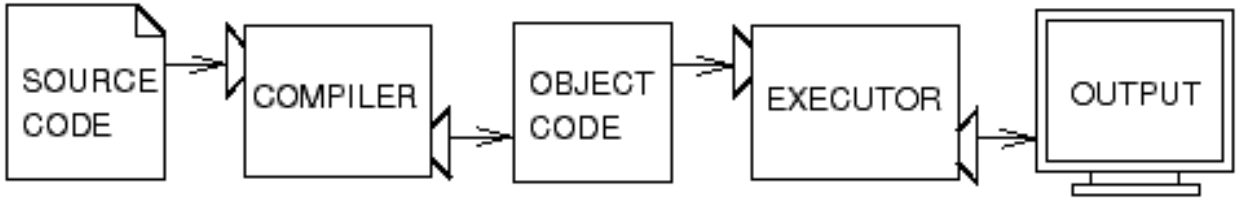
2 Trendy, aneb v čem a proč vyvíjet

Programovací jazyky

PROFINIT
new frontier group

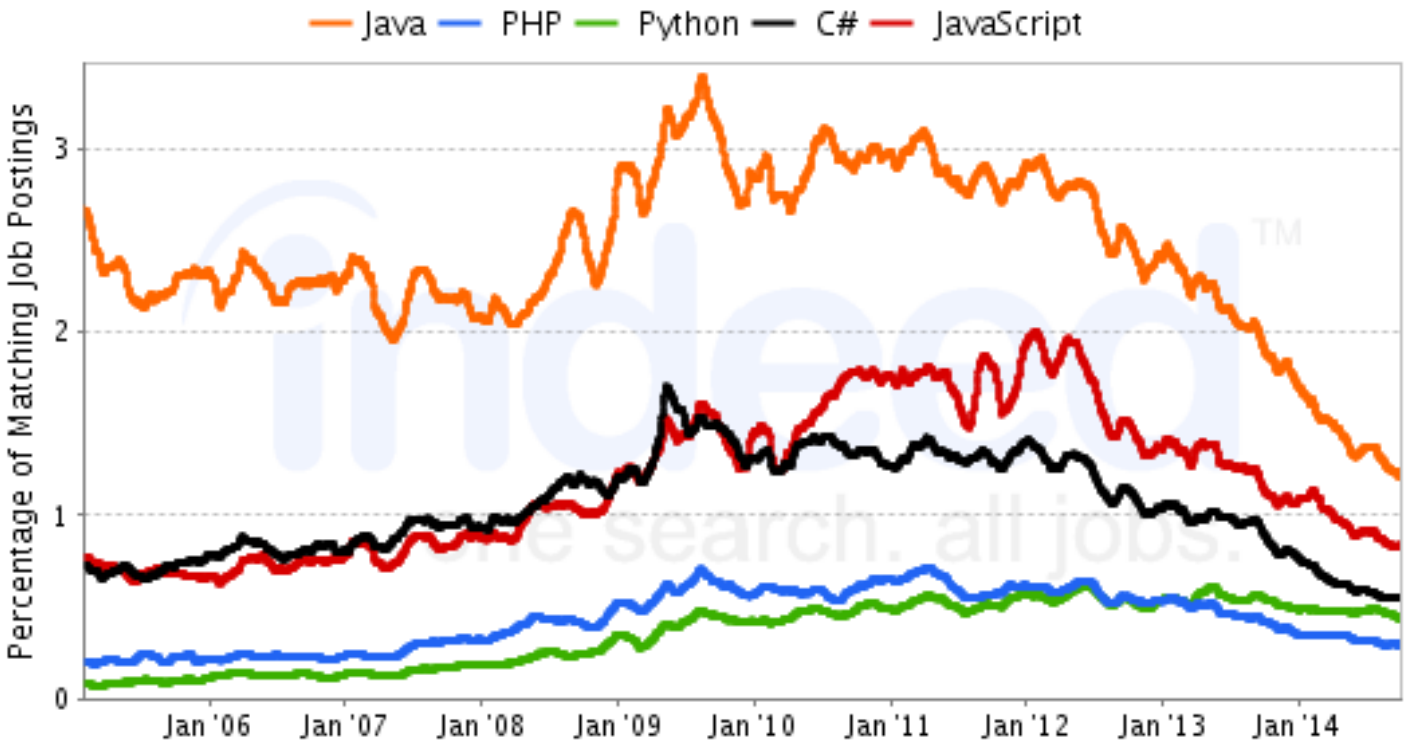
- Interpretované / Skriptovací
  - Běží všude, dynamické typování
  - Interpretace něco stojí

- Kompilované
  - Nutno kód „zprocesovat“ – kompilace
  - Běží na HW, optimalizace instrukcí, kontrola kódu kompilací



Popularita jazyků dle IEEE 2014

Popularita jazyků dle „indeed.com“



- Krize 08/09
  - Dobíhaly projekty, vypovídaly se smlouvy outsource (Indie, apod.)
  - Stabilizace týmů vývoje

Od nuly vs. Knihovny vs. Frameworky

PROFINIT
new frontier group

- Od nuly
  - Dobré pro učení, špatné pro produktivitu
  - Přebíráme na sebe odpovědnost za údržby „boiler plate“ – platformy
  - Proti principům YAGNI, best-tool

- Knihovny
  - Kompromisní řešení
  - Sami vybíráme funkcionalitu, kterou chceme využít
  - Kombinační a kompatibilní peklo

- Framework
  - Pro produktivitu nejlepší, ale vyšší cena „learning curve“
  - Musíme vhodně zvolit na startu
  - Vnucuje nám architekturu, postupy
  - Implementujeme a napojujeme na připravená místa

Co se používá v enterprise?

PROFINIT
new frontier group

- Java /.NET
  - Jazyky s důvěrou
  - Dostupná pracovní síla
  - Rychlé, stabilní, mnoho technologií a frameworků k integraci
- Java Enterprise Edition (JEE)
  - Speciální rozšíření pro potřeby enterprise od 1999
  - API a runtime prostředí (aplikační server)
  - Komponentová architektura
  - Princip „convention over configuration“
    - Pokud splníme konvence, nemusíme konfigurovat
    - Pojmenování tříd (beany, listenery)
  - Jasně definované API
    - ORM
    - Distribuované a vícevrstvé architektury
    - Webové služby
  - Frameworky JEE (Spring, apod.) staví na JEE

Co obsahuje JEE

PROFINIT
new frontier group

- javax.servlet.*
  - API pro zpracování HTTP požadavků (včetně template enginu JSP)
- javax.faces.* a javax.faces.component.* (tohle je JSF)
  - API pro tvorbu uživatelského rozhraní pomocí komponent
- javax.ejb.* (alternativou je např. Spring Framework)
  - API pro implementaci backendu na principu byznys procesů
  - Persistence, transakční zpracování, bezpečnost, task scheduling, asynchronní zpracování, meziprocesová komunikace, apod.
- javax.persistence.*
  - API kontrakt mezi providerem, spravovanými třídami a klientskou aplikací
- javax.transaction.* a javax.validation.*
  - Anotace pro definici transakčního zpracování a validací na backendu
  - Nutný provider (např. Hibernate)

Co se používá v enterprise?

PROFINIT
new frontier group

- Spring Framework
  - Alternativa k JEE
  - Lepší udržovatelnost a testovatelnost
  - Core (dependedy injection, integrace), Security, Integration, WebFlow

- Frontendové frameworky postavené na JSF
  - MyFaces, Trinidad, RichFaces, PrimeFaces

- Hibernate a myBatis
  - Persistence dat

Co se bude používat v enterprise ?

PROFINIT
new frontier group

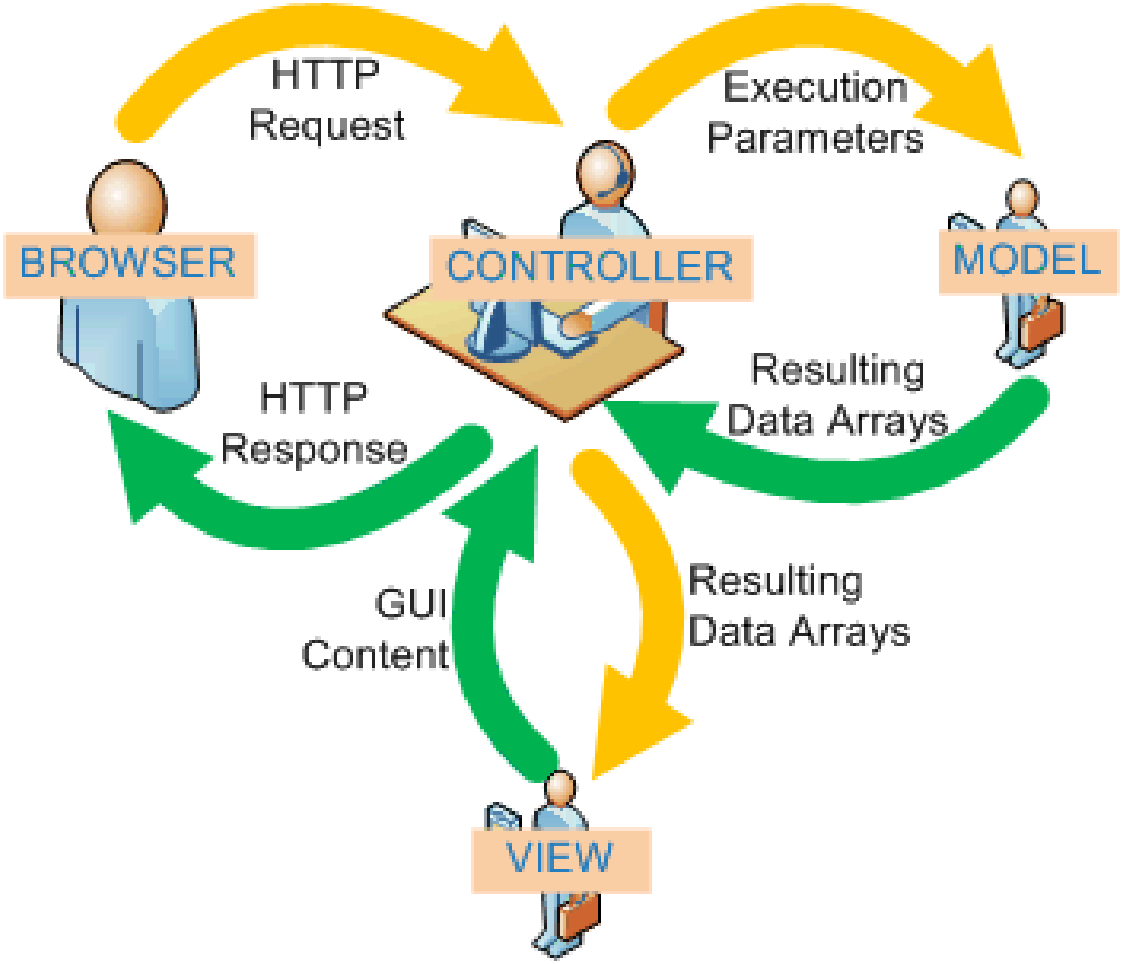
- Spring Framework
  - Zatím nemá rozumnou konkurenci
  - Stále drží nejvyšší laťku

- Hibernate a myBatis
  - Data jsou nejdůležitější část byznysu

- Frontendové frameworky na JavaScriptu
  - Lepší uživatelský komfort
  - Rychlejší odezva
  - Integrace na backend přes WS / REST

3 Frontendové frameworky

Architektura MVC

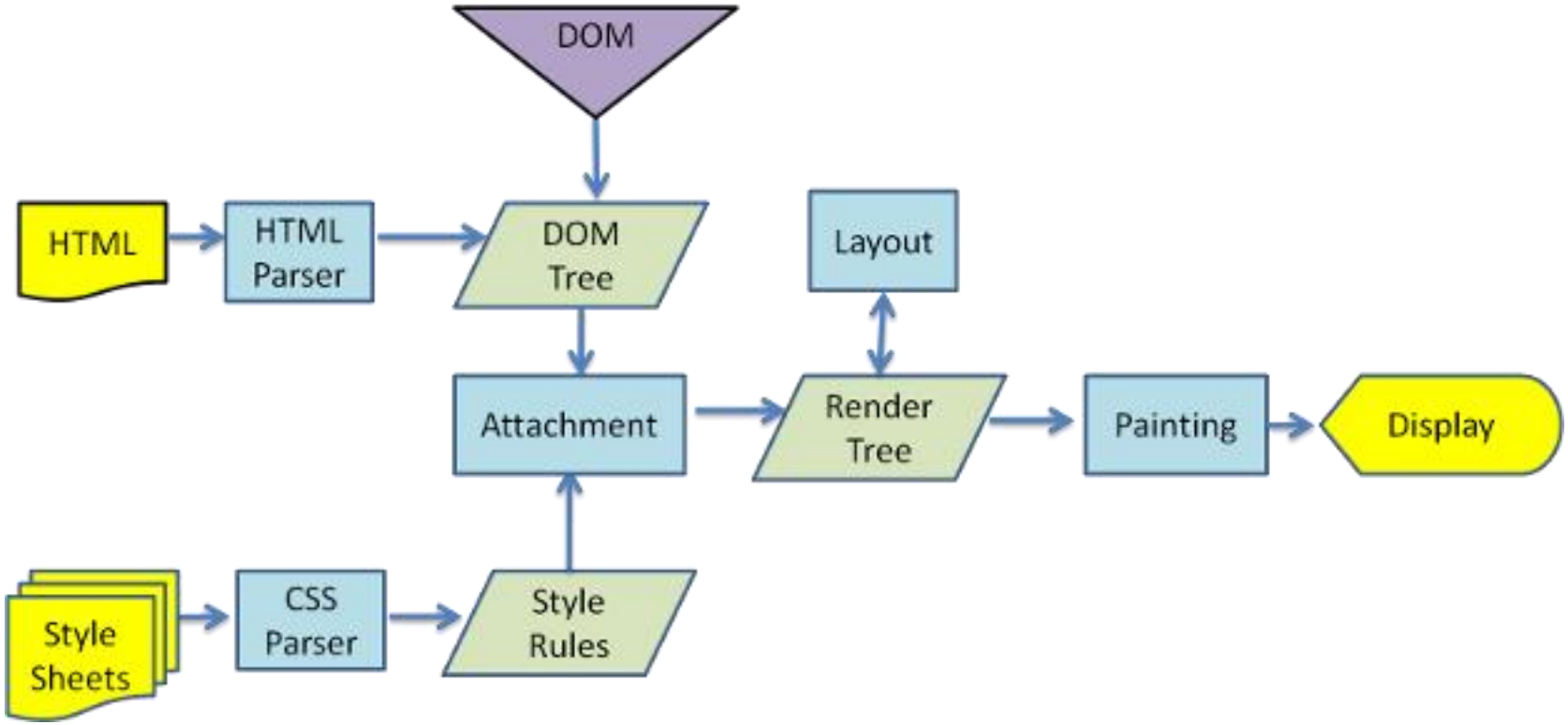


Model

- Závislost na „doméně“
- Reprezentace informací a operací
  - Data
  - Operace na datech
- Např. doména „pojišťovnictví“
  - Data
    - Smlouva
    - Klient
  - Operace smlouva
    - Založení/Změna/Zrušení smlouvy
    - Pojistné plnění na základě smlouvy
  - Klient
    - Založení

View

- Prezentace informací
- Web vždy HTML a CSS interpretované browserem



Controller

PROFINIT
new frontier group

- Logika aplikace
  - Reakce na události
  - Zajišťuje změny v modelu a view
- Standardně controller na serveru, ale může být i na klientovi
  - Na klientovi typicky u JavaScript frameworků

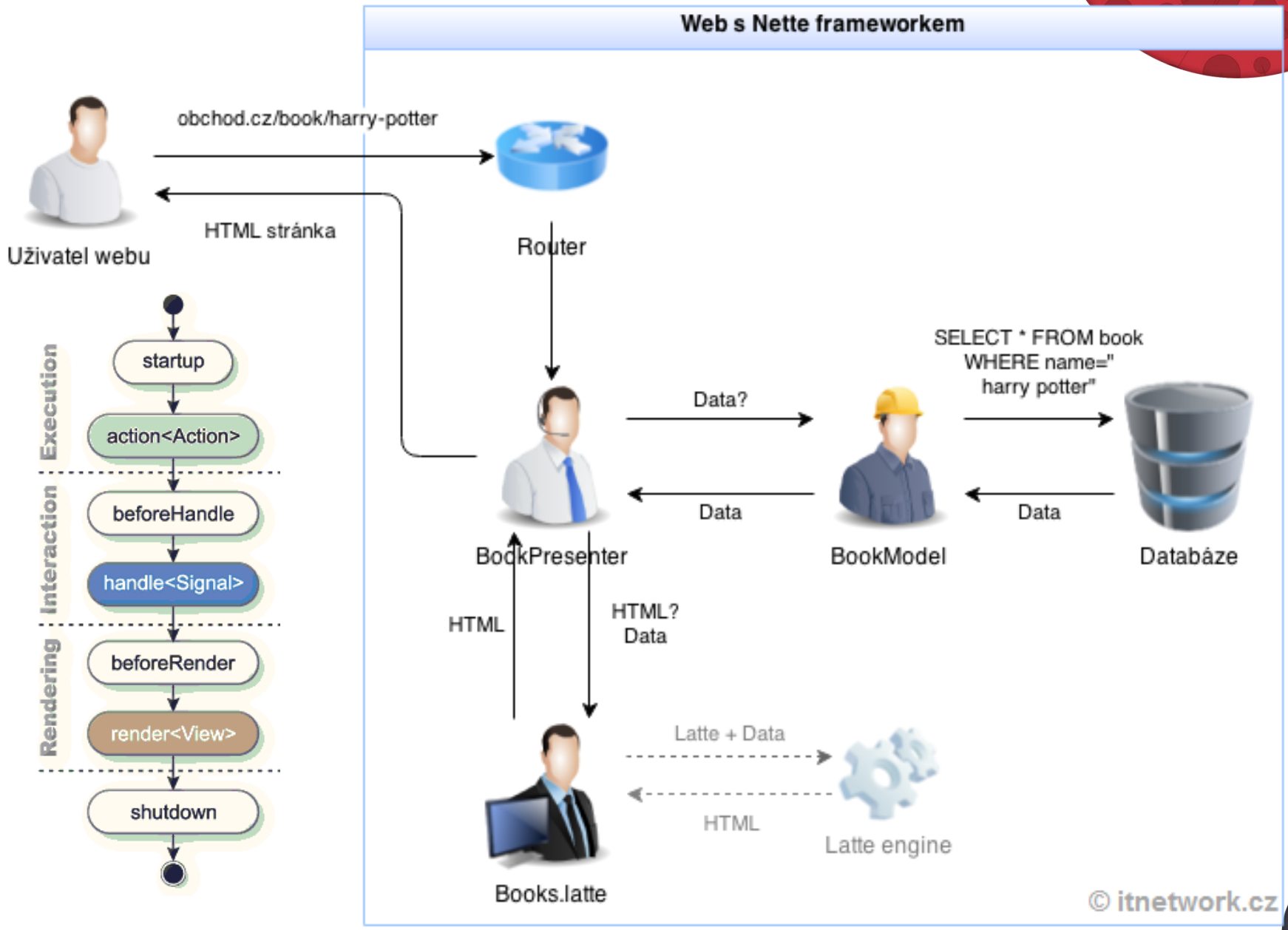
	JEE framework	JavaScript framework
Model	Databáze	Databáze
View	Template/komponentový engine	Přímo HTML, CSS, JavaScript
Controller	Beany, service	JavaScript třídy
Komunikace C ⇔ M	Persistence, WS	REST
Komunikace C ⇔ V	Parametry template	JavaScript

Controller

PROFINIT
new frontier group

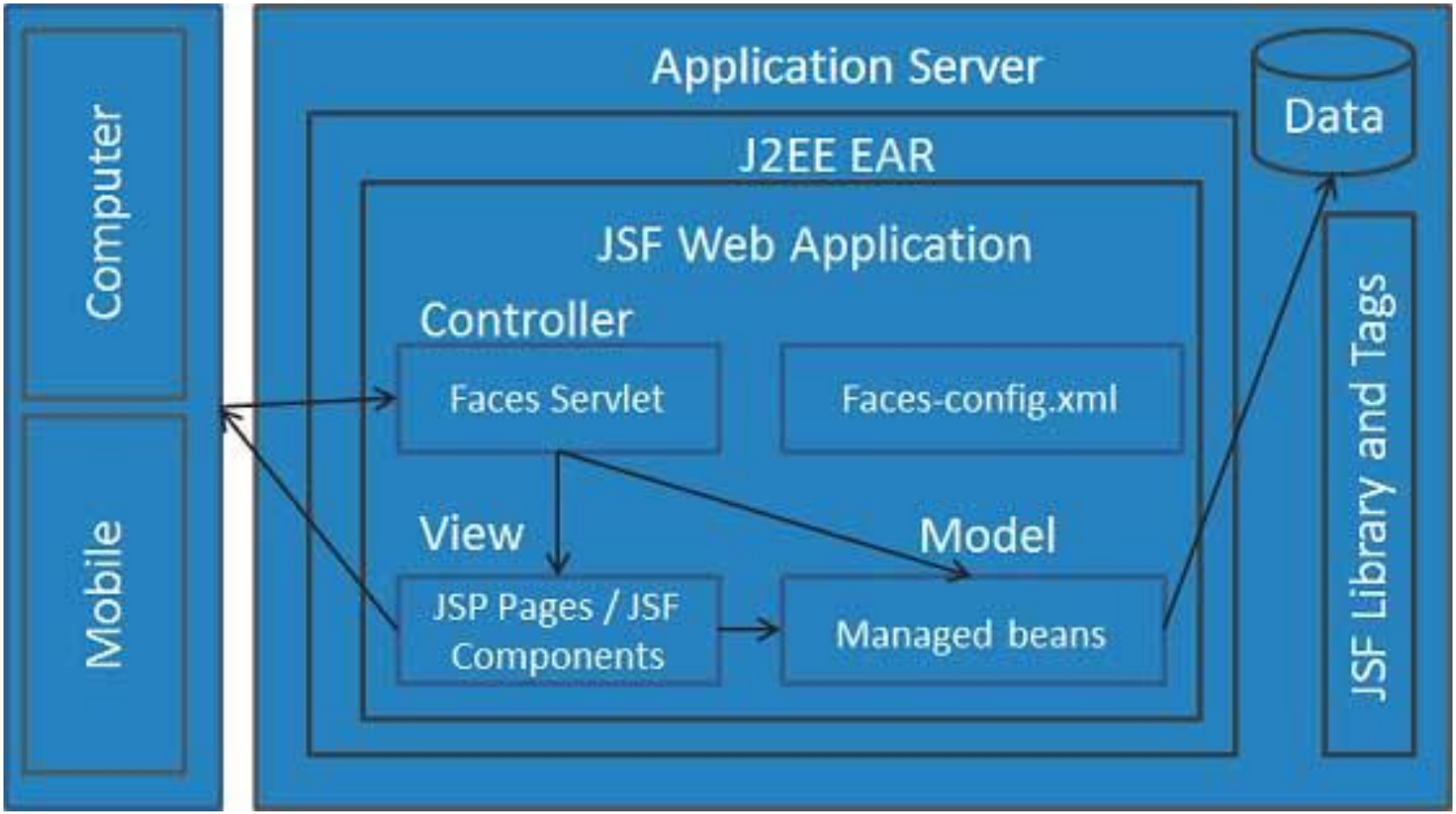
- Na serveru
  - Dobře prozkoumaná oblast, implementováno
  - Bezpečnost
  - Validace, formátování, apod. plně pod kontrolou
  - Overhead zpracování (zatěžujeme naše „železo“)
- Na klientovi
  - Relativně nová záležitost
  - Vyšší uživatelský komfort a rychlost odezvy
  - Nutná podpora u klienta (závislost na prohlížeči např. při interpretaci JavaScriptu)
- Ideální je kombinace
  - Vymazlený JavaScript-powered frontend
  - Serverový backend s validacemi, logikou, persistencí
    - Volání přes REST nebo WS

Koncept Nette (PHP)



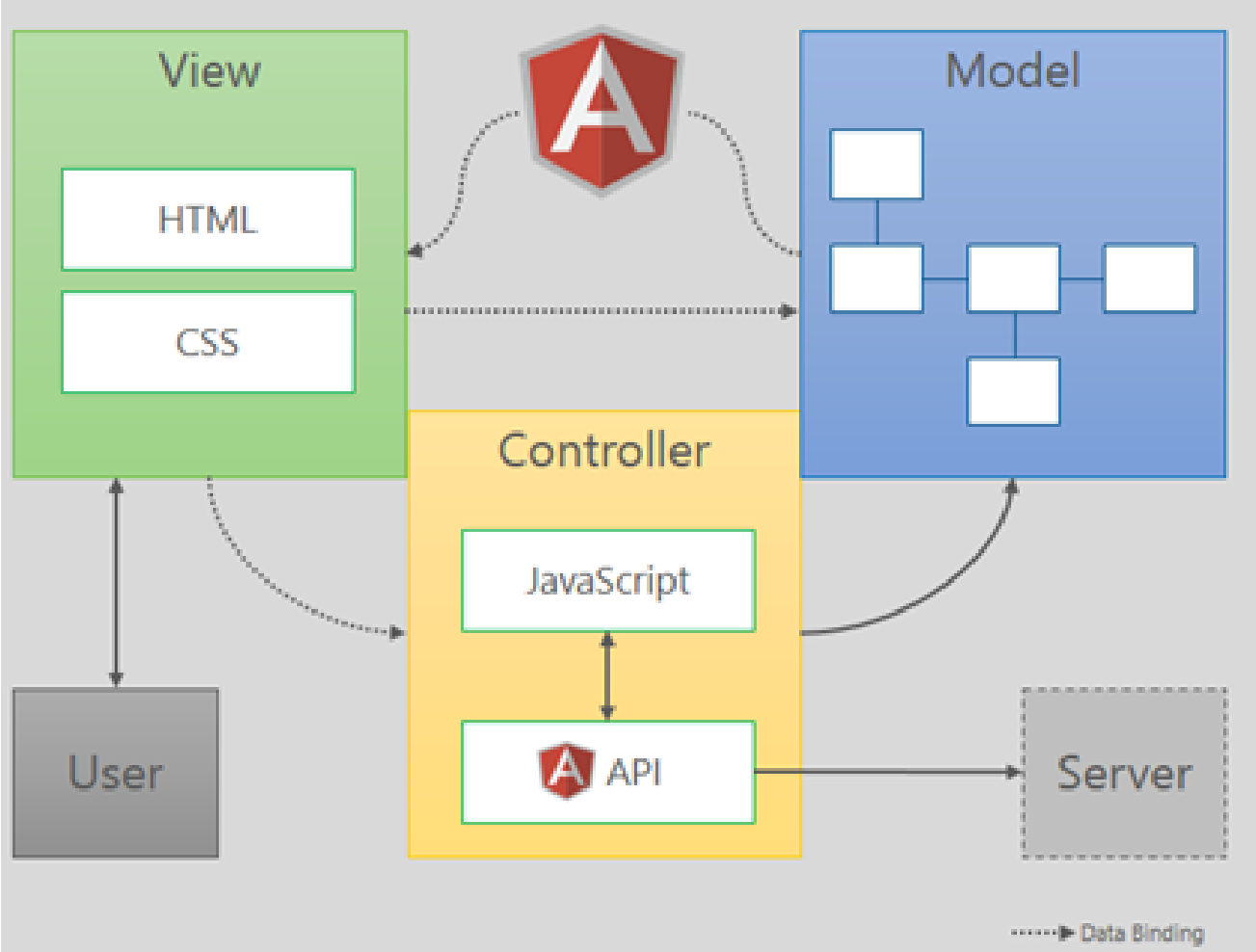
Koncept JSF (Java)

PROFINIT
new frontier group



Koncept AngularJS (JavaScript)

PROFINIT
new frontier group

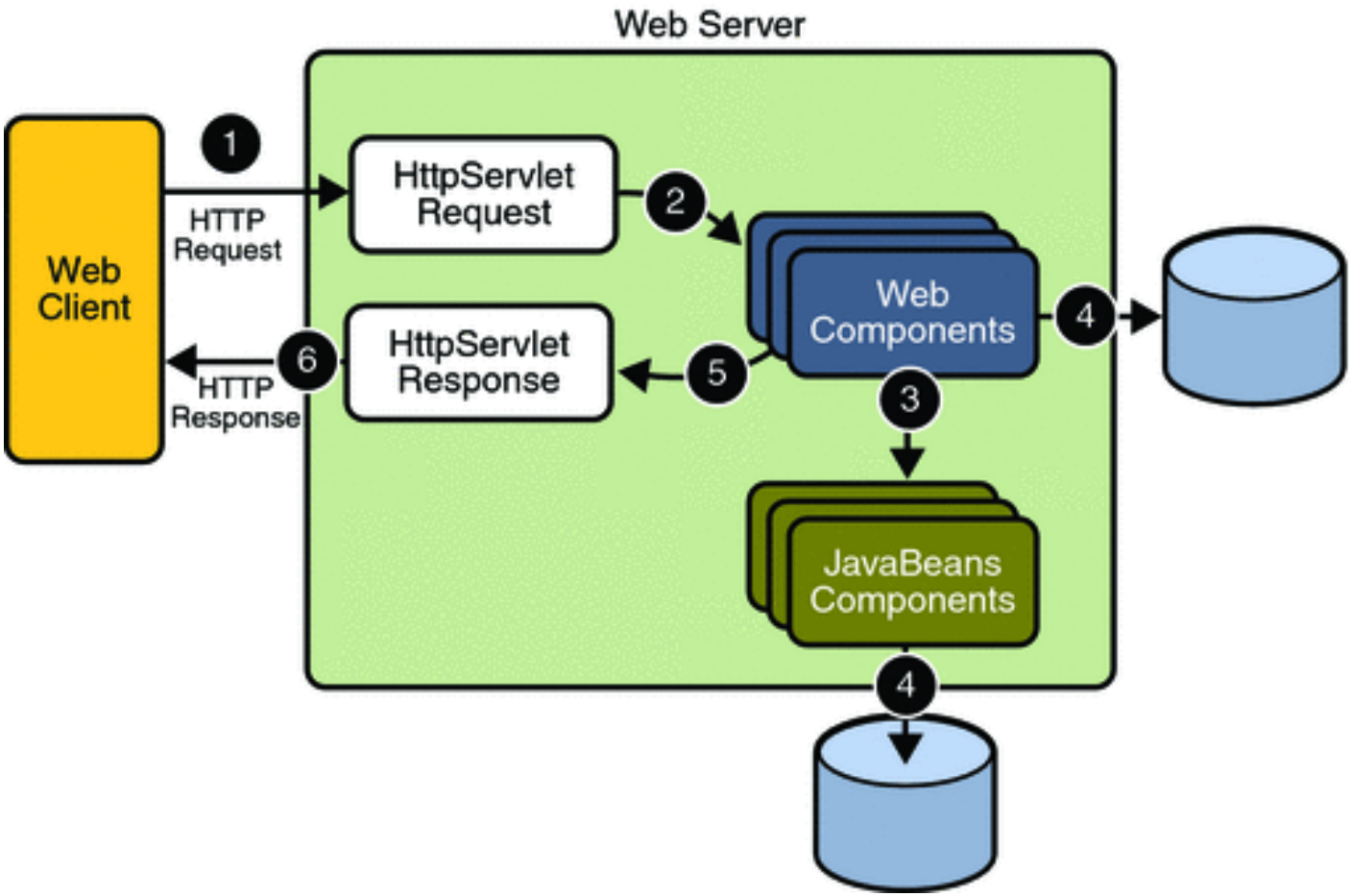


4 Java Server Faces

HTTP request/response v JEE

PROFINIT
new frontier group

- Request/Response zpracováný HTTP servlet



Co je tedy JSF ?

PROFINIT
new frontier group

- Framework pro komponentový vývoj webových aplikací – součást JEE
  - Předdefinované UI komponenty
  - Událostmi řízený model
    - Eventy
    - Listenery
    - Navigace
  - Rozšiřitelnost třetími stranami
    - Komponenty, validátory, filtry, konvertery
    - Změny v životním cyklu zpracování requestu
- JSF je oficiální „view“ technologie JEE, ale jde jen o specifikaci (dříve JSP)
  - Implementace typicky rovnou doplňuje JSF
  - RichFaces, PrimeFaces, IceFaces
    - Chytřejší komponenty
    - Více javascriptu
    - Lépe stylované ve výchozím nastavení
    - Nové funkčnosti nad JSF

Is JSF dead ?

PROFINIT
new frontier group

- Proč není doporučován?
  - Z historických důvodů s ohledem na JSF 1.2
  - Stavovost (session scope)
  - Abstrakce nad HTTP/HTML (což se některým nelíbí)

- Mají pravdu?
  - JSF 2.0 je příjemnější na práci (facelety, kompozice komponent)
  - Stavovost není na závadu, když se s tím počítá
  - Lepší abstrakce, než ladit elementární funkcionalitu a vlastní komponenty
- Důvody, proč použít
  - Stabilní v čase (verze nevychází tak často, spíše se stabilizuje funkčnost)
  - Nemusíte vymýšlet kolo
  - Oddělení prezentační a byznys logiky
  - Integrovatelné se širokou škálou frameworků (např. Spring Framework)

Historický vývoj webových aplikací (ve světě Javy)

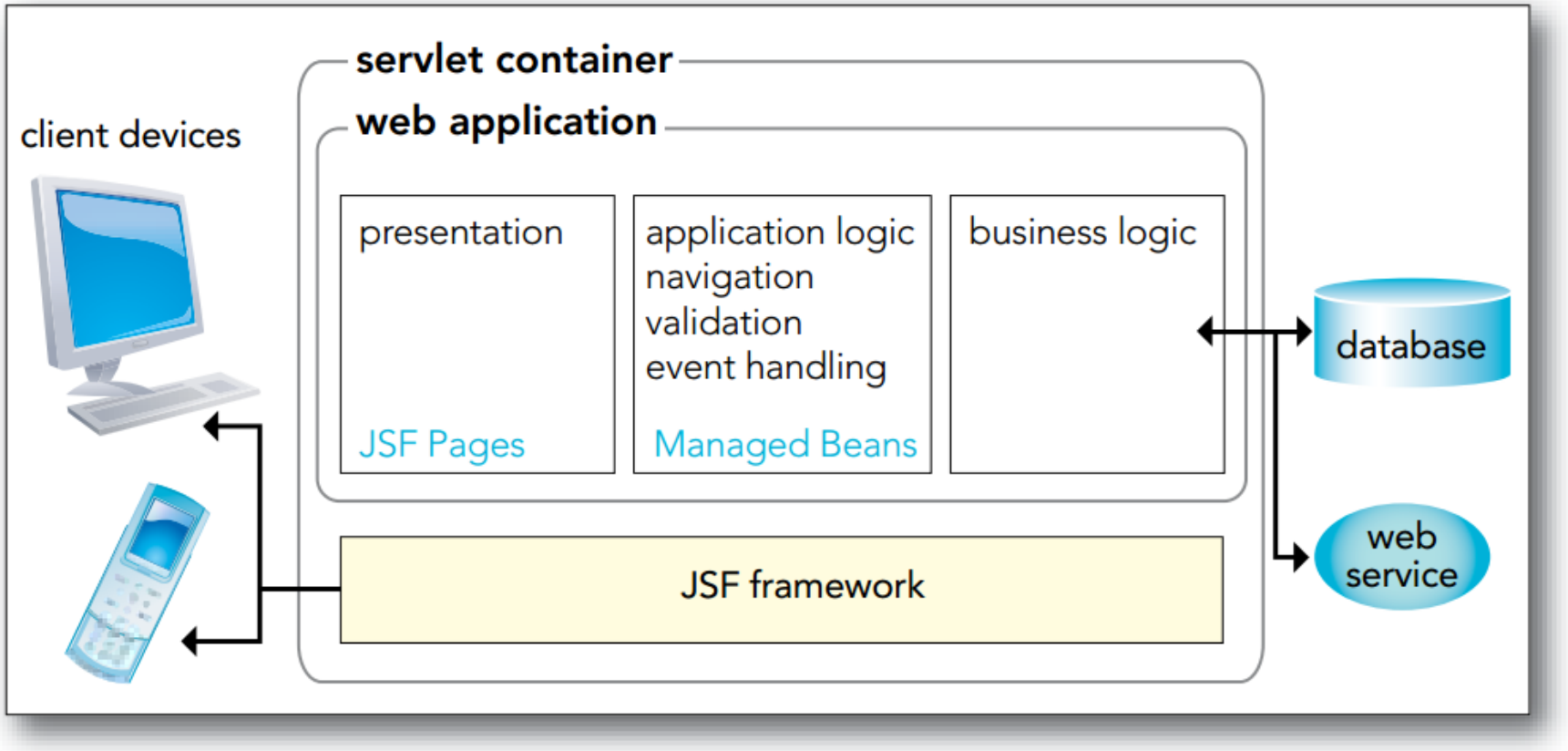
- Servlet
  - Request se zpracujne v kódu, ven jde string
- JSP
  - Template engine pro generování DOM
- Struts, apod.
  - Chytré propojení Servlet a JSP
- JSF 1.2
  - Komponentový koncept nad Servlet a JSP
- JSF 2
  - Komponentový koncept zobecněn a osvobozen od JSP
    - (X)HTML a JSF komponenty
    - Kompozice stránky
    - Komponenty snadno pomocí kompozice existujících komponent

Co nám JSF v JEE dává ?

PROFINIT
new frontier group

- MVC framework (komplet)
  - Template engine (facelety, XHTML)
  - Beany – aplikační logika, binding
- Komponenty
  - Předpřipravené standardní komponenty
  - Podpora ve všech prohlížečích (včetně IE6)
- Formátování, konverze, validace a error handling
- Navigace – flow
  - Složitější procesní flow
  - Stavovost
- Podpora lokalizace, přístupnosti

Co nám JSF v JEE dává ?



Základní koncept (konfigurace)

PROFINIT
new frontier group

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns="http://java.sun.com/xml/ns/javaee"
        xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
        xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
        http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
        id="WebApp_ID" version="2.5">

        <display-name>JavaServerFaces</display-name>

        <!-- Welcome page -->
        <welcome-file-list>
                <welcome-file>faces/hello.xhtml</welcome-file>
        </welcome-file-list>

        <!-- JSF mapping -->
        <servlet>
                <servlet-name>Faces Servlet</servlet-name>
                <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
                <load-on-startup>1</load-on-startup>
        </servlet>

        <!-- Map these files with JSF -->
        <servlet-mapping>
                <servlet-name>Faces Servlet</servlet-name>
                <url-pattern>*.xhtml</url-pattern>
        </servlet-mapping>
</web-app>
```

Základní koncept (stránka)

PROFINIT
new frontier group

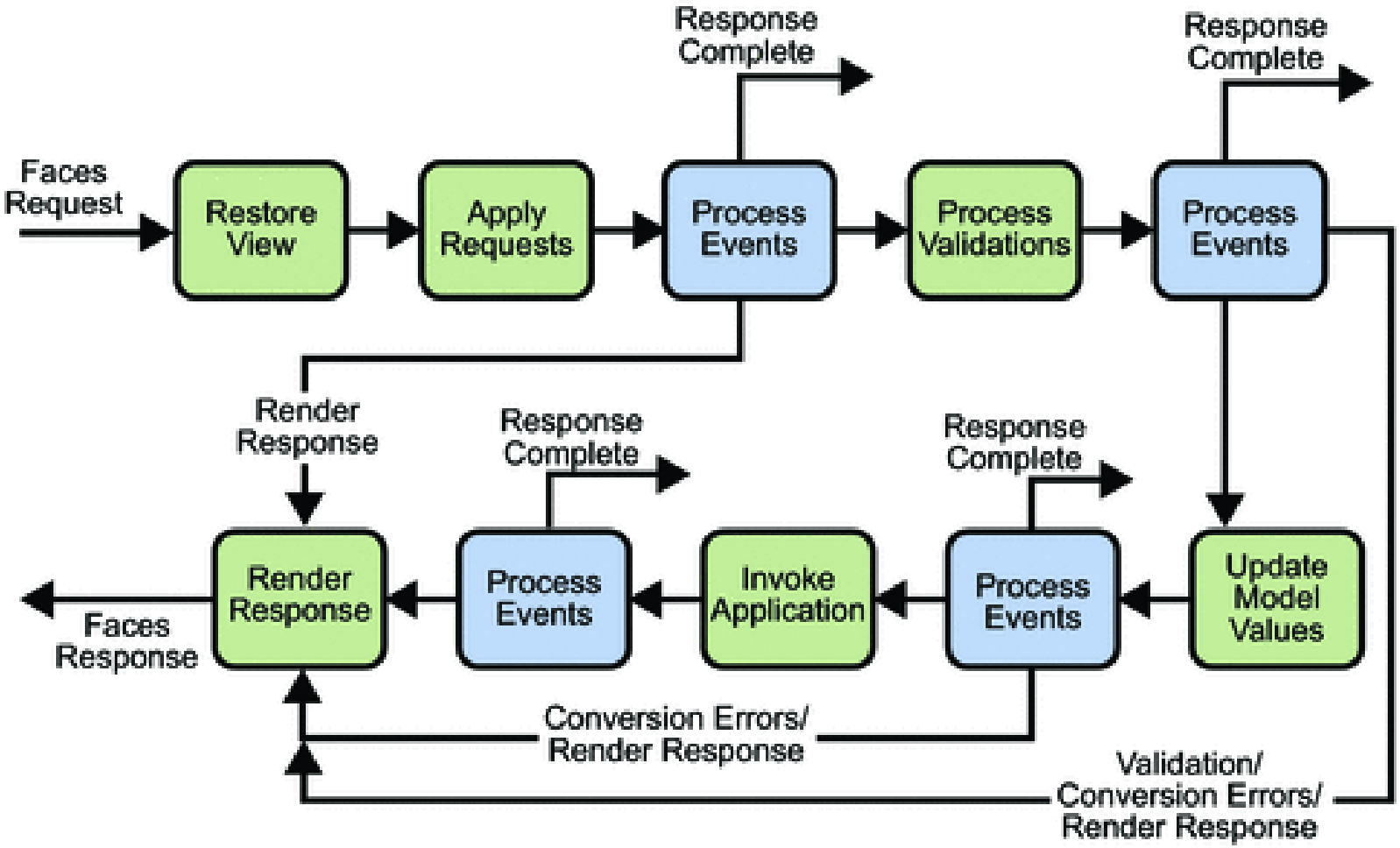
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html">

    <h:head>
        <title>JSF 2.0 Hello World</title>
    </h:head>
    <h:body>
        <h3>JSF 2.0 Hello World Example - hello.xhtml</h3>
        <h:form>
           <h:inputText value="#{helloBean.name}"></h:inputText>
           <h:commandButton value="Welcome Me" action="welcome"></h:commandButton>
        </h:form>
    </h:body>
</html>
```

Základní koncept (beana)

PROFINIT
new frontier group

```
package com.mkyong.common;

import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;
import java.io.Serializable;

@ManagedBean
@SessionScoped
public class HelloBean implements Serializable {

	private static final long serialVersionUID = 1L;

	private String name;

	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
}
```

Životní cyklus requestu v JSF



5 JSF – komponenty

Nativní komponenty

PROFINIT
new frontier group

- Nativní jsou součástí JSF
  - xmlns:f=http://java.sun.com/jsf/core
    - view, ajax, param
    - selectItems, selectItem
    - actionListener, valueChangedListener
    - validator, validateBean, validateLength, validateRequired
    - converter, convertDateTime, convertNumber
  - xmlns:h=http://java.sun.com/jsf/html
    - body, head, form
    - inputText, inputSecret, inputHidden
    - panelGrid, panelGroup, dataTable
    - selectOneListBox, selectManyListBox
    - messages, message
    - commandButton, commandLink
  - xmlns:ui=http://java.sun.com/jsf/facelets
    - insert, include, composition
    - define, param
    - repeat

Nativní komponenty

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core" xmlns:ui="http://java.sun.com/jsf/facelets">
    <h:head>
        <h:outputStylesheet library="css" name="table-style.css"  />
    </h:head>
    <h:body>
        <h1>JSF 2 dataTable example</h1>
                <h:dataTable value="#{order.orderList}" var="o"
                        styleClass="order-table"
                        headerClass="order-table-header"
                        rowClasses="order-table-odd-row,order-table-even-row">

                        <h:column>
                                <!-- column header -->
                                <f:facet name="header">Order No</f:facet>
                                <!-- row record -->
                                <h:commandLink action="order.detail(o.orderNo)">
                                        #{o.orderNo}
                                </h:commandLink>
                        </h:column>
                        <h:column>
                                <f:facet name="header">Product Name</f:facet>
                                #{o.productName}
                        </h:column>
                </h:dataTable>
    </h:body>
</html>
```

Kompozitní komponenty

PROFINIT
new frontier group

- Kompozitní vytváříme ze standardních přes facelety
  - xmlns:composite=http://java.sun.com/jsf/composite
    - composite:interface
    - composite:attribute
    - composite:implementation
  - Uložení mezi resources a použití přes namespace balíku (nemusíme taglib)
    - webapp/resources/mojeKomponenty
    - xmlns:mojeKomponenty="http://java.sun.com/jsf/composite/mojeKomponenty"

```
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:composite="http://java.sun.com/jsf/composite">
    <composite:interface>
        <composite:attribute name="anything" />
        <composite:attribute name="anyAction" method-signature="java.lang.String action()" />
    </composite:interface>

    <composite:implementation>
        #{cc.attrs.anything}
        <h:commandLink action="#{cc.attr.anyAction}" value="My action"/>
    </composite:implementation>
</html>
```

Validace

PROFINIT
new frontier group

- Validace probíhají vesměs na serveru, ale deklarují se na různých místech
  - V XHTML šabloně
    - validator
    - validateLength, validateRequired, validateRegex
    - validateBean
  - Anotacemi na beaně na property (JSR 303)
    - @NotNull, @NotBlank
    - @Min, @Max, @Length
    - @Email, @Pattern, @DateTimeFormat, @Past, @ValidateDateRange
    - vlastní (např. @RodneCislo, @Ico)
  - V kódu listeneru nebo action
    - Pro cross-field validace
    - Byznys validace
    - Zachycení výjímek (např. middleware, ORM, apod.)
    - ValidatorException(message)
      - Severity, ID komponenty
      - Summary, detail

Podpora lokalizace

PROFINIT
new frontier group

- faces-config.xml

```
<application>
   <resource-bundle>
       <base-name>com.corejsf.messages</base-name>
       <var>msgs</var>
   </resource-bundle>
</application>
```

- messages.properties a messages_xx.properties
  - WEB-INF/classes/com/corejsf/messages.properties (výchozí)
    - goodbye=Good bye
  - WEB-INF/classes/com/corejsf/messages_cz.properties
    - goodbye=Nashledanou

- XHTML template

```
<h:outputText value="#{msgs.goodbye} !">
```

Podpora AJAX

PROFINIT
new frontier group

- Široká podpora od JSF 2.0
  - Nemusíme psát JavaScript
    - Vše potřebné generuje JSF
    - Možnost přidat do libovolné komponenty
- JSF lifecycle pracuje s podstromy modelu
  - Samotná komponenta
  - Odkazované komponenty (pro render fázi a parse fázi)
  - Metody v beaně vrací „void“ nebo „return(null)“ vyvolající rerender

```
<h:commandButton value="Execute" action="order.execute()">
    <f:ajax render="id1 id2" execute="id3 id4"
            event="onclick" onevent="javaScriptHandler"/>
</h:commandButton>
```

Knihovny a rozšíření

PROFINIT
new frontier group

- Mohou přidávat funkcionality
  - Více user-friendly komponenty
  - Podpora CSS templatování (themes, styles)
  - Responsive-design
    - Automatické přizpůsobení vzhledu stránky dostupným rozměrům
    - 1 implementace pro web, tablet, mobil
    - trend
  - Podpora mobilního vývoje
    - Komponenty využívající spec. vlastností/funkčností mobilu
- Nemusí být vzájemně kompatibilní
  - Není dobré kombinovat frameworky
    - Často odlišný koncept
    - Problémy na úrovni generovaného JavaScriptu
  - Většinou si plně vystačíme s jediným frameworkem
    - Kompozice zařídí zbytek

6 JSF – Beany

Co je beana ?

PROFINIT
new frontier group

- Inversion of Control
  - Sami třídu nespravujeme, necháme to na frameworku
- JavaBeans jsou třídy definované konvencí
  - Camel-case název
  - Zapouzdřené property
    - private scope
    - přístup výhradně přes getX/isX a setX
  - Serializovatelné
  - Konstruktor bez argumentů
- Konfigurace
  - Anotacemi
  - XML popisem

Co je beana ?

PROFINIT
new frontier group

- Beana JSF je přístupná v UI
  - Model = interní property s daty
  - Controller = action, actionListener
  - Přístup přes expression language
    - Uzavřen do #{expression}
- Od JSF 2.0
  - Rozšířený EL o metody s parametrem
    - #{navBean.orderDetail(15).detail}
  - Definice anotacemi
    - Preferované řešení
- Dependency injection
  - Aplikační kontejner spravuje beany
    - Dependency injection v beanách
  - Vytvářeny dle potřeby
    - Scope beany

Konfigurace anotacemi

PROFINIT
new frontier group

```
package com.project.common;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;
import java.io.Serializable;

@ManagedBean
@SessionScoped
public class HelloBean implements Serializable {

        private static final long serialVersionUID = 1L;

        private String name;

        public String getName() {
                return name;
        }

        public void setName(String name) {
                this.name = name;
        }
}
```

Konfigurace XML

```
<!-- web.xml -->
...
   <context-param>
      <param-name>javax.faces.CONFIG_FILES</param-name>
      <param-value>WEB-INF/manage-beans.xml</param-value>
    </context-param>
...

<!-- manage-beans.xml -->
<?xml version="1.0" encoding="UTF-8"?>
<faces-config
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-facesconfig_2_0.xsd"
    version="2.0">
    <managed-bean>
          <managed-bean-name>helloBean</managed-bean-name>
          <managed-bean-class>com.project.common.HelloBean</managed-bean-class>
          <managed-bean-scope>session</managed-bean-scope>
     </managed-bean>
</faces-config>
```

Anotace

PROFINIT
new frontier group

- @ManagedBean
  - Označuje třídu určenou ke správě aplikačním kontejnerem
    - Výchozí název je název třídy s 1. písmenem lowercase
    - Změna názvu přes atribut „name“
  - Lazy vs. Eager loading
    - Atribut „eager“
    - **True** = beana bude vytvořena ještě před jejím použitím (dle scope)
    - **False** = beana bude vytvořena až při prvním použití
- @ManagedProperty
  - Dependency injection na úrovni bean dle jména přes setter

```
@ManagedBean
@SessionScoped
public class HelloBean implements Serializable {
        @ManagedProperty(value="#{message}")
        private MessageBean messageBean;

        //must povide the setter method
        public void setMessageBean(MessageBean messageBean) {
                this.messageBean = messageBean;
        }
}
```

Scope

PROFINIT
new frontier group

- Paměťový kontext v aplikačním kontejneru

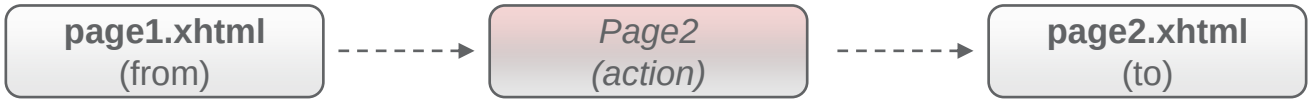
Scope	Rozsah platnosti
@RequestScoped	Beana je platná po dobu zpracování HTTP request/response. Vzniká před zpracováním requestu a zaniká po odeslání response.
@ViewScoped	Beana je platná po dobu existence 1 view. Vzniká před zpracováním requestu na daném view a zaniká přechodem na jiné view.
@SessionScoped	Beana je platná po dobu trvání session. Vzniká před zpracováním requestu při založení nové session a zaniká při invalidaci session.
@ApplicationScoped	Beana je platná po dobu běhu aplikace. Vzniká při prvním requestu na danou beanu (pro eager=„true“ při spuštění aplikace) a zaniká při ukončení aplikace. Standardně plní roli služny (ve SpringFramework je to singleton).
@NoneScoped	Beana je platná po dobu vyhodnocení 1 EL. Vzniká při zahájení zpracování expression a zaniká po jejím vyhodnocení.
@CustomScoped	Beana je platná po dobu existence reference ve speciální mapě. Scope je spravován manuálně. Vlastní implementace mapy, kterou je potřeba registrovat. Stejně tak je potřeba zaregistrovat vlastní EL resolver.

7 JSF – Navigace (flow)

Vestavěná podpora navigace

PROFINIT
new frontier group

- V JSF 1.2 navigace definicí v faces-config.xml




```
<navigation-rule>
   <from-view-id>page1.xhtml</from-view-id>
   <navigation-case>
       <from-outcome>page2</from-outcome>
       <to-view-id>/page2.xhtml</to-view-id>
   </navigation-case>
</navigation-rule>
```

- Od JSF 2.0 možnost využít implicitní navigace

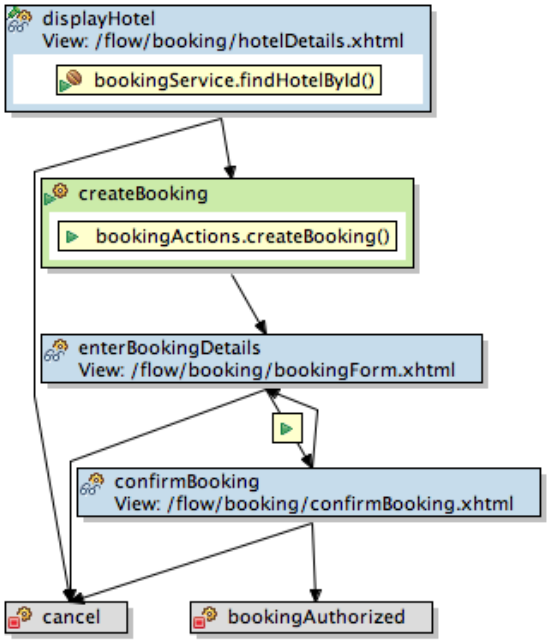
```
<h:commandButton action="page2" value="Move to page2.xhtml" />

<h:commandButton action="#{pageController.moveToPage2}"
                 value="Move to page2.xhtml by managed bean" />
```

```
@ManagedBean
@SessionScoped
public class PageController implements Serializable {
        public String moveToPage2(){
            return "page2"; //outcome
        }
}
```

Složitější navigace např. přes SWF

PROFINIT
new frontier group

- Lze využít mechanismus faces-context.xml
  - IDE podporují generování stavového/přechodového diagramu
  - Navigace roztříštěna na více místech
    - Controller (beana)
    - XML popis přechodů
- Standardně se využívá framework třetí strany
- Top hraje Spring WebFlow
  - Možnost tvorby složitých přechodových automatů
    - viewState, actionState, decisionState, subflowState
    - více přechodů z 1 stavu
    - Konfigurace v XML
    - testovatelnost unit testy, včetně logiky
  - Controller je plně přesunut do definice přechodového automatu
  - Model je čistě POJO beana



Exception hadler

- Globální exception handler – handling chain
  - Nutno vytvořit exception handler přes factory, rozšíření ExceptionHandlerFactory
    - Jednoduchý objekt vracející novou instanci
    - Definuje chain (parent handler)
  - Implementace handleru rozšiřuje ExceptionHandlerWrapper
    - Obsluhuje vybrané výjimky
    - Část může delegovat na výchozí handler
    - Obsluha může navigovat aplikaci např. na error page

```
<factory>
        <exception-handler-factory>
            com.projekt.exception.CustomExceptionHandlerFactory
        </exception-handler-factory>
</factory>
```

Exception hadler

PROFINIT
new frontier group

```
public class CustomExceptionHandler extends ExceptionHandlerWrapper {
    private static final Logger log = Logger.getLogger(CustomExceptionHandler.class.getCanonicalName());
    private ExceptionHandler wrapped;
    ...

    @Override
    public void handle() throws FacesException {

        final Iterator<ExceptionQueuedEvent> i = getUnhandledExceptionQueuedEvents().iterator();
        while (i.hasNext()) {
            ExceptionQueuedEvent event = i.next();
            ExceptionQueuedEventContext context =
                    (ExceptionQueuedEventContext) event.getSource();

            Throwable t = context.getException(); // get exception from context

            final FacesContext fc = FacesContext.getCurrentInstance();
            final Map<String, Object> requestMap = fc.getExternalContext().getRequestMap();
            final NavigationHandler nav = fc.getApplication().getNavigationHandler();

            try {
                // EXCEPTION HANDLING
            } finally {
                //remove it from queue
                i.remove();
            }
        }
        getWrapped().handle(); //parent handle
    }
}
```

8 Specifické části JSF

Vestavěné statické funkce

PROFINIT
new frontier group

- Řetězcové funkce (namespace fn = http://java.sun.com/jsp/jstl/functions)
  - boolean contains(String, String)
  - int length(Object)
  - String toLowerCase(String) a String toUpperCase(String)
  - String trim(String)

```
<?xml version="1.0" encoding="UTF-8"?>
<facelet-taglib
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
                        http://java.sun.com/xml/ns/javaee/web-facelettaglibrary_2_0.xsd"
    version="2.0">
    <namespace>http://example.com/functions</namespace>

    <function>
        <function-name>contains</function-name>
        <function-class>com.example.Functions</function-class>
        <function-signature>
            boolean contains(java.util.Collection, java.lang.Object)
        </function-signature>
    </function>
</facelet-taglib>
```

Implicitní objekty

PROFINIT
new frontier group

- Dostupné v EL
  - Cookie
  - facesContext
    - Objekt celého aplikačního kontextu
    - Přístupný i přímo z Javy
  - Jednotlivé scope
    - requestScope, viewScope, sessionScope, applicationScope
    - prakticky jde o mapu
  - Session
    - Informace o konkrétní session
    - Spravuje aplikační server

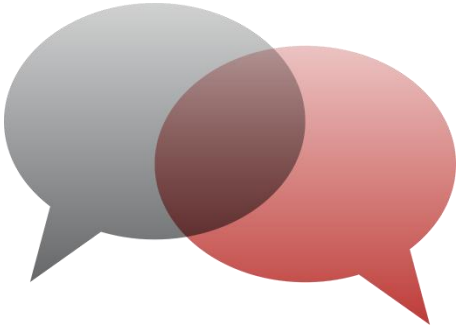
Obecný request handling

PROFINIT
new frontier group

- JSF proces zpracování HTTP requestu životním cyklem
  - PhaseListener – možno vlastní implementace
    - Rozšiřitelnost
    - Napojení rozšíření třetích stran
  - Možno kompletně přepsat mechanismus zpracování requestu
  - Typicky místo, kde se napojují ostatní frameworky
    - Správa bean
    - Správa navigace
    - Validační mechanismy
    - Alternativní komponentové frameworky

9 PrimeFaces showcase

PROFINIT
new frontier group



Diskuze

PROFINIT
new frontier group

Děkuji za pozornost!

Společnost PROFINIT je členem nadnárodní skupiny New Frontier Group, která je leadrem v oblasti digitální transformace organizací a firem ve střední a východní Evropě. S více než 2000 zaměstnanci v 17 zemích patří mezi deset největších poskytovatelů ICT služeb v celém CEE regionu a řadí se ke špičce v oblasti vývoje software na zakázku, data managementu, datových skladů a business intelligence.

PROFINIT má řadu významných zákazníků z finančního a telekomunikačního sektoru, utilit a státní správy. Společnost se primárně zaměřuje na konzultační služby v oblasti digitální transformace, technologické služby a outsourcing. Podle údajů IDC (2012) patří PROFINIT mezi 5 největších firem v oblasti vývoje software na zakázku v České Republice a je držitelem řady dalších ocenění.

Profinit, s.r.o., Tychonova 2, 160 00 Praha 6, +420 224 316 016, www.profinit.eu