

Linguagem de Programação para Web

Aplicações Web

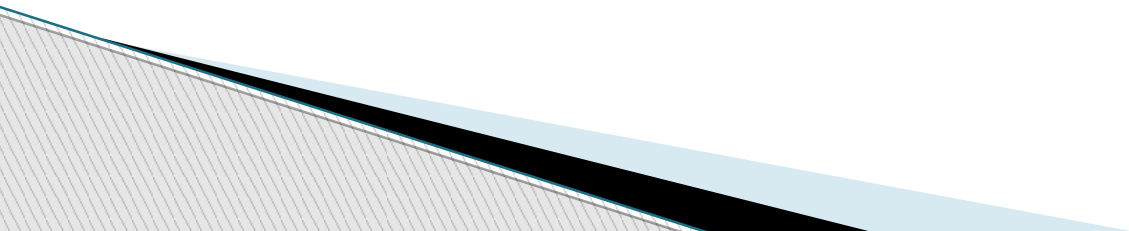
Prof. Tales Bitelo Viegas

<https://fb.com/ProfessorTalesViegas>

The bottom of the slide features a decorative graphic consisting of several overlapping, wavy lines in shades of gray and light blue, creating a modern, abstract background element.

Aplicação Web

- ▶ Uma aplicação que é executada em um navegador
- ▶ Tudo que executa em um navegador pode ser considerado uma aplicação?



Aplicação Web



MENU

ge



CONFRONTO

Q BUSCAR



ENCERRADO

C. Brasileiro
AVA
FLU 1
0

C. Brasileiro
SAN
CFC 3
0

C. Brasileiro
CAP
SPO 1
1

C. Brasileiro
VAS
JEC 0
0

C. Brasileiro
PON
FLA 1
0

C. Brasileiro
GOI
CAM 0
0

C. Brasileiro
SAO
COR 1
1

C. Brasileiro
CRU
PAL 2
1

C. Brasileiro
GRE
INT 5
0

C. Brasileiro
CHA
FIG 2
2



CAMPEONATO BRASILEIRO RODADA 17

3ª

GRÊMIO

Giulliano, Fernandinho, Réver (contra), Luan (2)



5 × 0



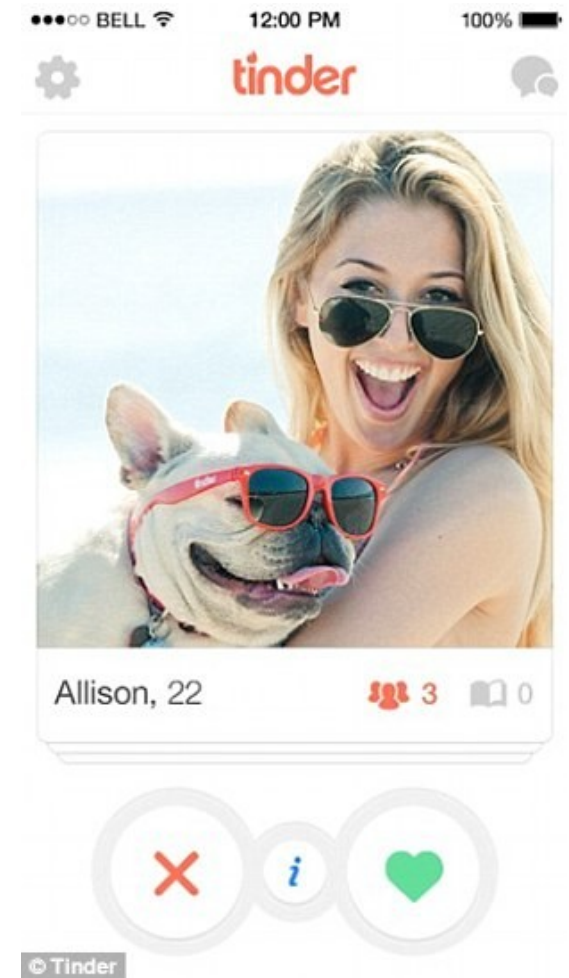
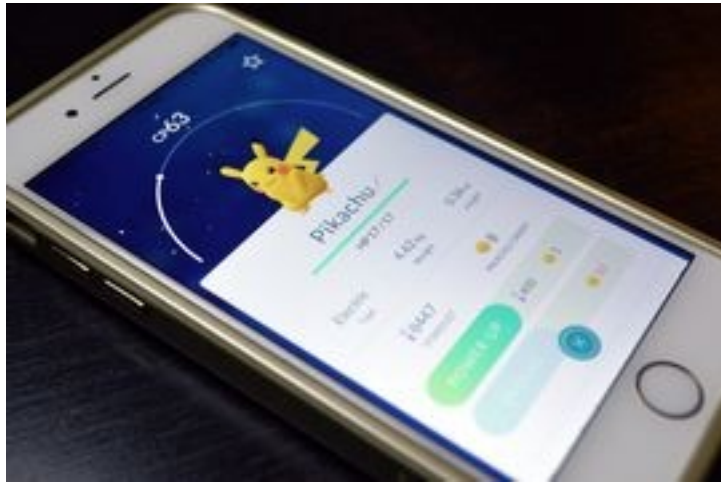
INTERNACIONAL

5ª

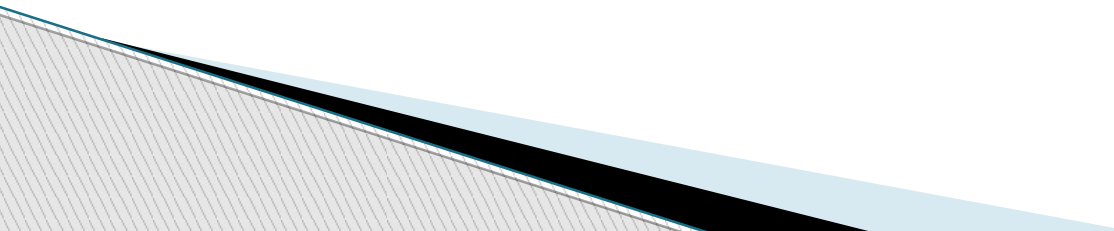
Grêmio massacra Inter com goleada histórica na Arena e retorna ao G-4

Tricolor aplica 5 a 0, salta cinco posições e afunda o maior rival de técnico interino em crise

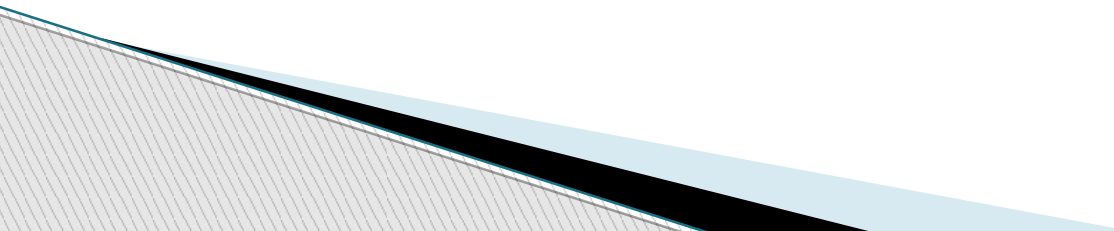
Aplicação Web



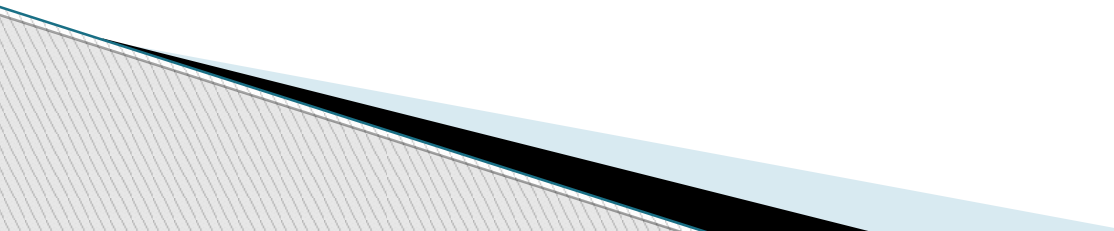
Vantagens

- ▶ Não requerem procedimentos de instalação (apenas do navegador)
 - ▶ Geralmente requerem pouco espaço em disco
 - ▶ Atualizações são enviadas para todos os clientes
 - ▶ Integração simples com outros sistemas (como e-mail, buscadores, etc.)
 - ▶ Execução em múltiplas plataformas
 - ▶ Criação de conteúdos multimídia (HTML5)
 - ▶ Disponível em qualquer computador conectado
- 

Desvantagens

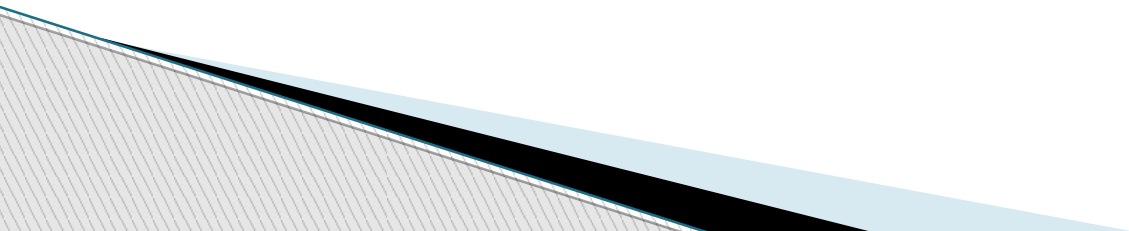
- ▶ Incompatibilidade entre navegadores
 - ▶ Necessita conexão dedicada*
 - ▶ Se a companhia que prevê a solução fecha por algum motivo, os dados são perdidos
 - ▶ * Temos outras soluções para este problema
- 

Tecnologias

- ▶ HTML
 - ▶ JavaScript
 - ▶ CSS
 - ▶ Ajax (XmlHttpRequest), XML, JSON
 - ▶ Flash
 - ▶ Browser Extensions
 - ▶ Frameworks
 - ▶ Tecnologias de Back-end
- 

Desktop vs Web

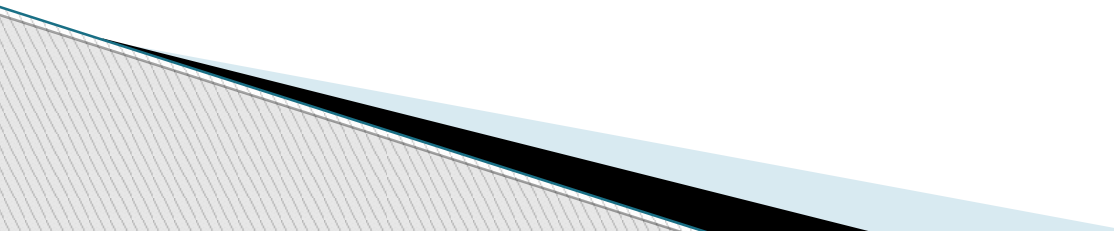
- ▶ Que aplicativo necessitamos ter instalado no nosso computador, além do navegador?
- ▶ E se o sistema operacional já é o navegador?



Chrome OS



A ARPANET

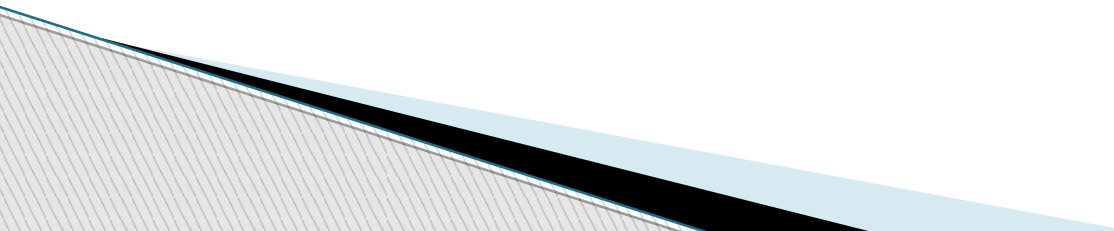
- ▶ Advanced Research and Projects Agency
 - ▶ Interligação das bases militares e departamentos de pesquisas norte-americanos em 1969
 - ▶ Dividiu-se em Milnet (Nipret) e Internet em 1983
 - ▶ 2.3 bilhões de dólares no orçamento americano de 2012
- 

A World Wide Web

- ▶ Concebida por Tim Bernes-Lee em 1989 (quando tinha 34 anos)
- ▶ Desenvolvida pelo CERN (*Conseil Européen pour la Recherche Nucléaire*)



A web é movida por

- ▶ Infraestrutura
 - ▶ Hardware
 - Servidores
 - Rede (cabos, switches, roteadores, etc.)
 - ▶ Software
 - Servidor
 - Cliente
 - ▶ Protocolos
 - HTTP (rede)
 - Protocolos de aplicação Web
- 

Endereço IP

- ▶ Endereço único de um dispositivo Web
- ▶ IPv4
 - 32 bits de endereçamento
 - Aproximadamente 4 bilhões de endereços
 - Ex: 200.134.12.10
- ▶ IPv6
 - 128 bits de endereçamento
 - $3,4 * 10^{38}$
 - 2001:12c0:0:100:c4ad:e8e7:27cf:1a3a

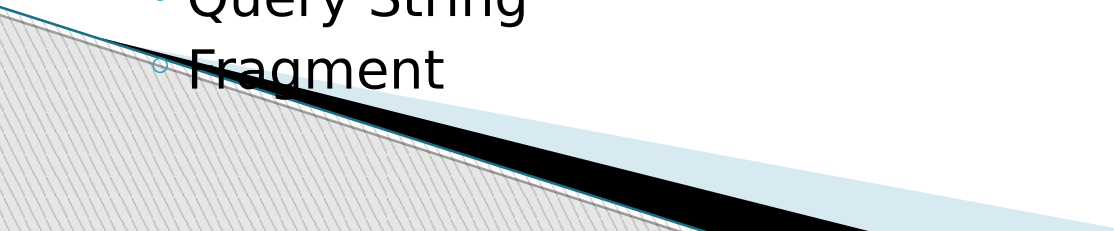
Serviço DNS

- ▶ DNS = Domain Name Service
- ▶ Faz a tradução de nomes para números IP
- ▶ Ex:
 - gravatai.ulbra.tche.br = 187.53.210.11
 - ▢ nslookup
 - ▢ host

URL

- ▶ Uniform resource locator
- ▶ É um nome associado a um endereço na Web

<http://user:senha@gravatai.ulbra.tche.br:80/portal/index.html?busca=teste#inicial>

- Protocolo
 - Usuário:Senha
 - Domínio
 - Porta
 - Caminho (path)
 - Query String
 - Fragment
- 

HTTP

- ▶ Protocolo
 - Regras e formatos para comunicação de dados
- ▶ HTTP = Hipertext Transfer Protocol
- ▶ Recupera recursos referenciados por uma URL

Software

- ▶ Navegadores



- ▶ Servidores



- ▶ Linguagens

Arquitetura Cliente/Servidor

- ▶ Cliente envia requisições e processa respostas
- ▶ Servidor recebe requisições, processa a requisição, envia resposta

Software Cliente

- ▶ Navegador (browser)
 - Composto de:
 - ▢ Um codificador/decodificador de HTTP
 - ▢ Um renderizador de HTML e CSS
 - ▢ Um interpretador de JavaScript
- ▶ Qualquer outra aplicação que compreenda o protocolo HTTP

Software Servidor

- ▶ Servidor Web
 - Exemplos: Apache, IIS, Nginx
- ▶ Composto de:
 - Codificador/decodificador HTTP
 - Processador de requisições
 - Ambiente de execução de linguagens

Arquitetura de Aplicações Web

- ▶ Decisões consistentes, aplicadas ao longo do desenho do **projeto**, a fim de garantir os **requisitos não-funcionais**.
 - Segurança
 - Escalabilidade
 - Disponibilidade
 - Portabilidade
 - Extensibilidade
 - Usabilidade

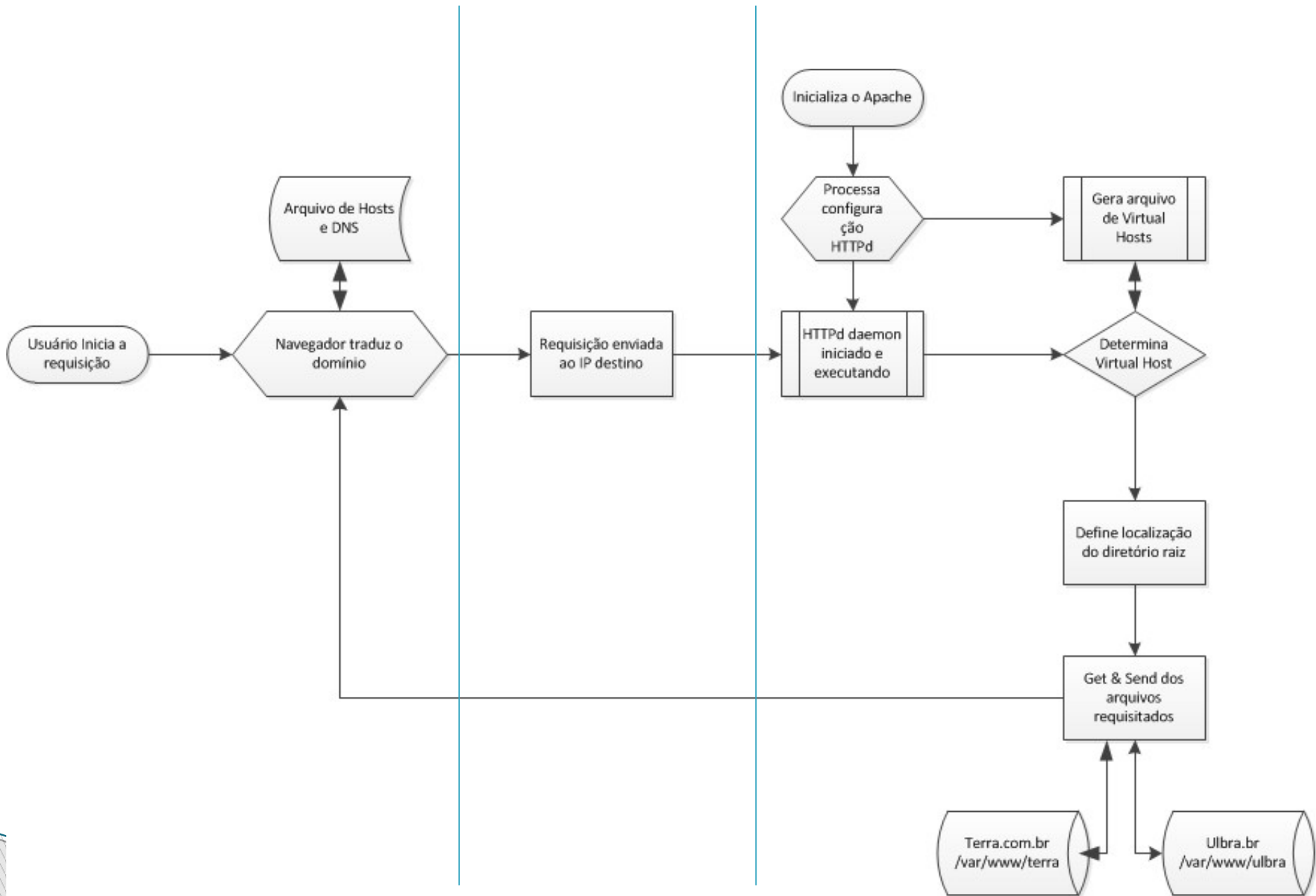
Arquitetura básica

- ▶ O básico: HTTP e URL



Arquitetura básica

- ▶ Requisitar um arquivo estático (que não precisa de um processamento no servidor) disponível em um servidor remoto
- ▶ Geralmente são páginas HTML, arquivos CSS, JavaScript, Imagens

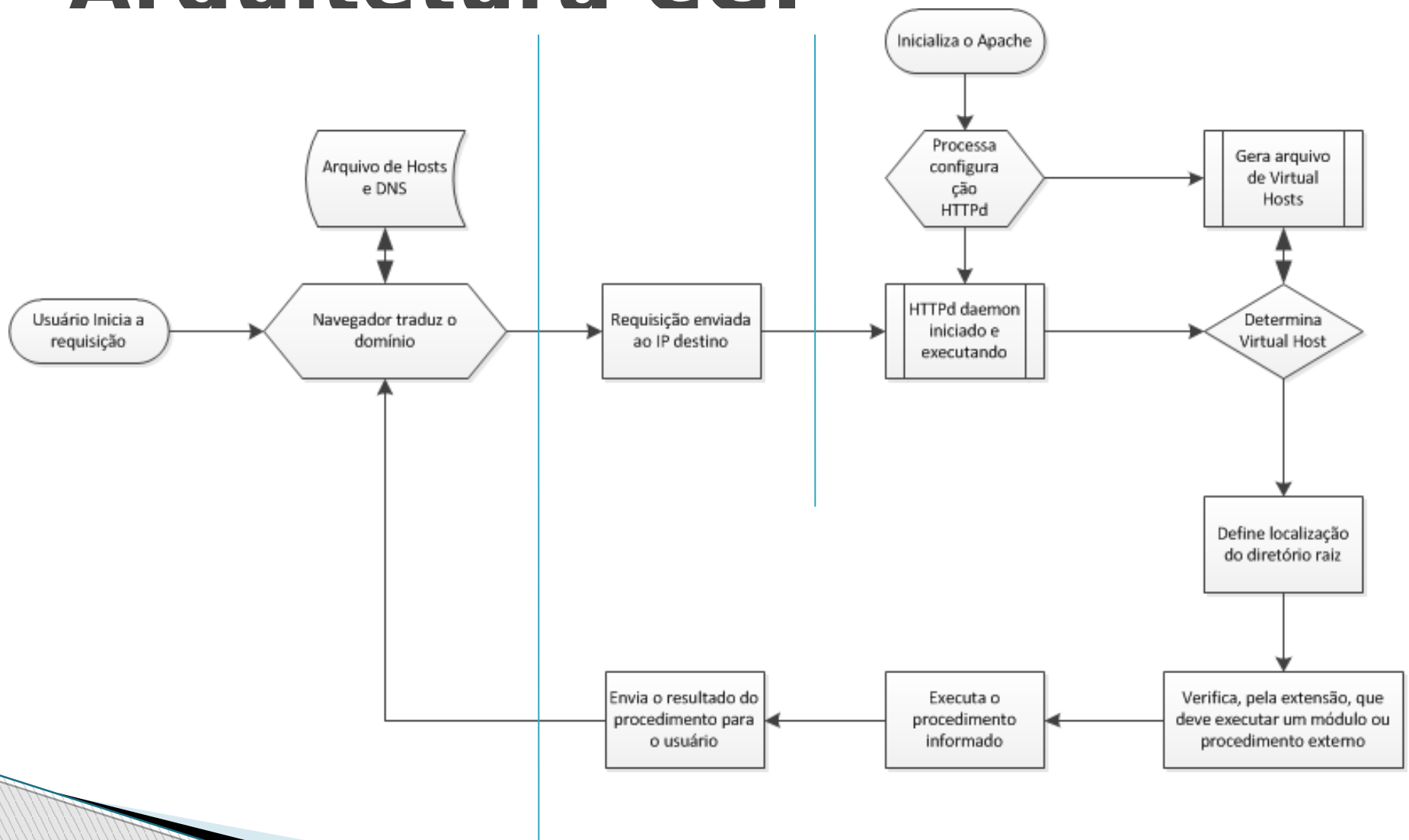


Arquitetura CGI

- ▶ Permitir a execução de programas no servidor, a partir de uma URL, e visualizar sua resposta através do navegador.
- ▶ Problema: não há um ambiente de execução do tipo sandbox



Arquitetura CGI



Arquitetura com Linguagem Script

- ▶ Utilizada para páginas dinâmicas
- ▶ Mistura código em uma linguagem de programação com código HTML
- ▶ Funciona da mesma forma que CGI, mas facilita a exibição

Arquitetura com Linguagem Script

```
<table>
```

```
<?
```

```
    while ($row = mysql_fetch_array($result)){
```

```
    ?>
```

```
        <tr>
```

```
            <td><?= $row['name'] ?></td>
```

```
            <td><?= $row['email'] ?></td>
```

```
        </tr>
```


```
<?
```

```
    } // end while
```

```
    ?>
```

```
</table>
```

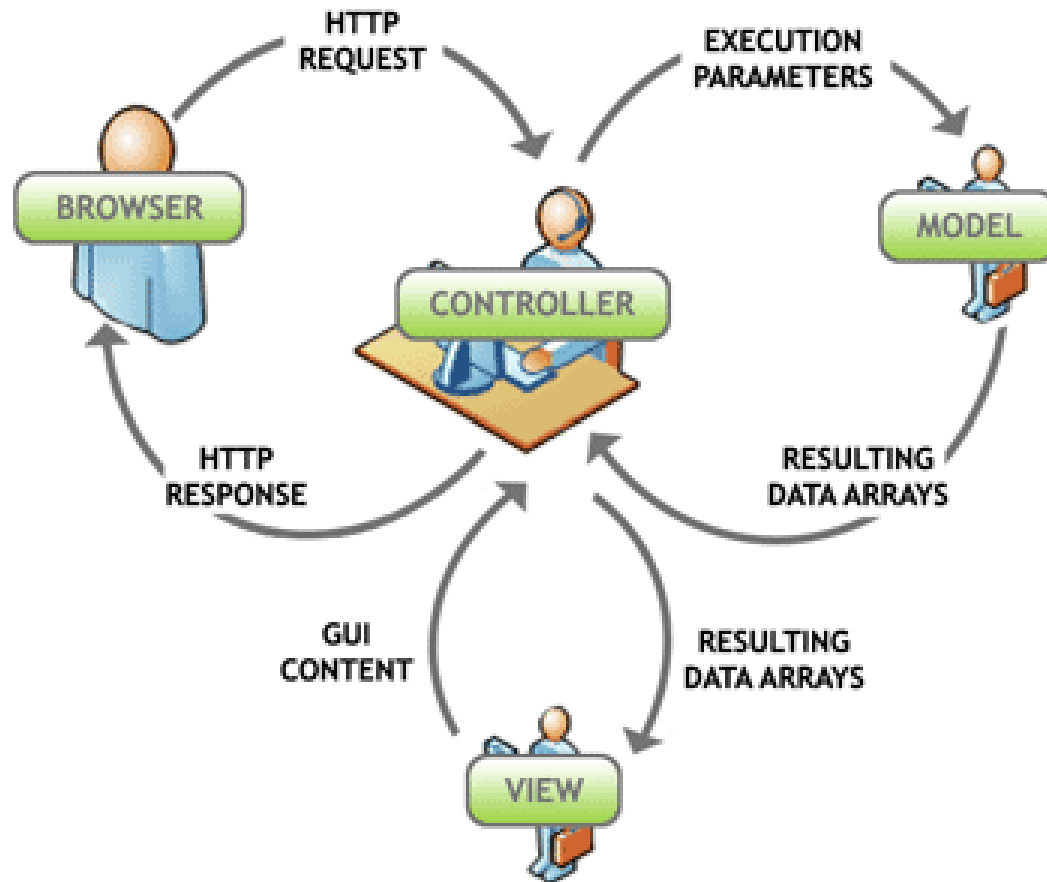
Arquiteturas MV(C ou P ou VM)

- ▶ Padrões que descrevem uma abordagem para desenvolvimento de software
 - ▶ Os módulos incluem:
 - *Model* – dados e comportamento
 - *View* – camada de apresentação
 - *C ou P ou VM* – lógica para “grudar as coisas”
 - ▶ Se baseiam em “Separação de Responsabilidades”
- 

Arquitetura MVC

- ▶ Model/View/Controller
- ▶ Descrita inicialmente em 1979 para Smalltalk (Xerox)
- ▶ O *controller* é a peça central que desacopla o Model e o View
- ▶ Fluxo de Controle
 - Evento de interação do usuário
 - Controller manipula o evento e converte para uma ação do usuário que o Model possa entender
 - Model gerencia o comportamento e dados do domínio da aplicação
 - Controller interage com o view para gerar a interface com o usuário


Arquitetura MVC



Arquitetura MVC

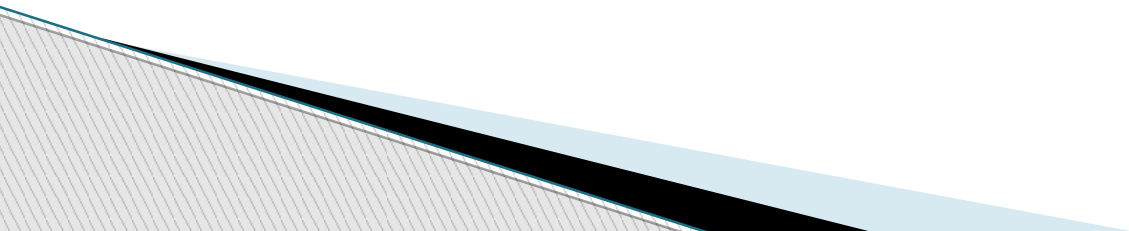
- ▶ Exemplos:
 - Perl: Catalyst, Mojolicious
 - PHP: Laravel, Symfony, Yii, Zend, CakePHP, Codeigniter
 - Java: Spring, JSF, Vaadin, GWT, Struts
 - Python: Plone, Django, TurboGears, Flask
 - Ruby: Rails, Sinatra
 - .Net: ASP.Net

Arquitetura MVP

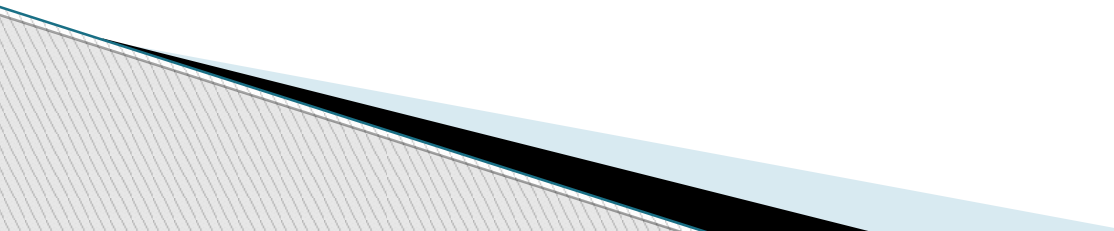
- ▶ Model/View/Presenter
 - ▶ Originou-se no início dos anos 90, derivado do MVC
 - ▶ Dois tipos de implementação:
 - Passive view
 - Supervising controller
 - ▶ Presenter assume a responsabilidade do MVC Controller
- 

Arquitetura MVP

- ▶ View é responsável por manipular os eventos da UI
- ▶ Model torna-se estritamente um modelo de domínio
- ▶ Mais voltado à interface com usuário



Arquitetura MVVM

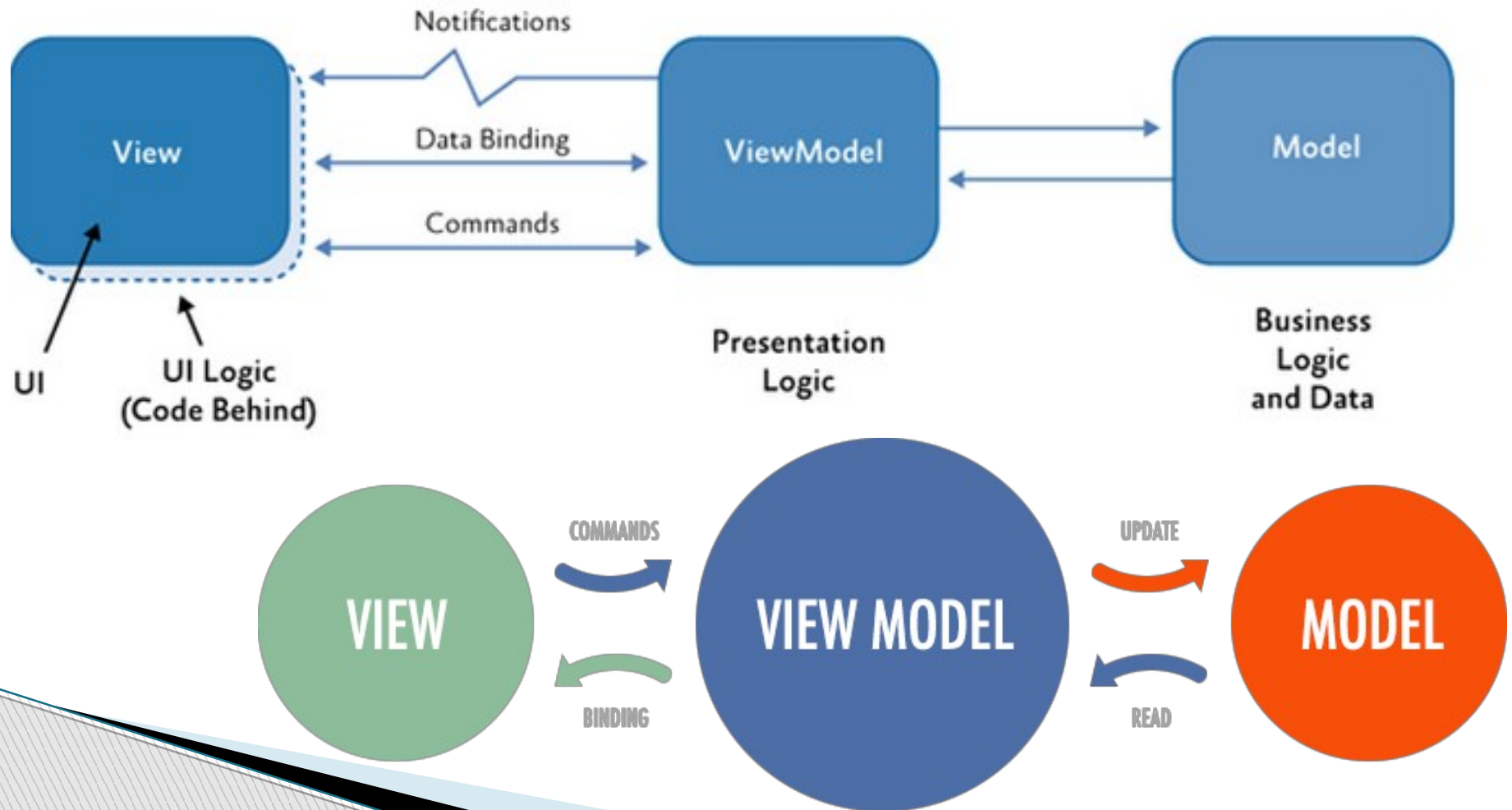
- ▶ Especialização do MVP, conhecida como modelo de apresentação
 - ▶ Construído primeiramente para o WPF e Silverlight
 - ▶ Model e View funcionam como no MVC
- 

Arquitetura MVVM

- ▶ ViewModel é um “Model da View”
 - Ela estende o Model com comportamentos (behaviors) que o View possa usar
 - Associação de dados (data-binding) entre View e Model
 - Passa comandos entre a View e o Model

Arquitetura MVVM

The MVVM classes and their interactions



Frameworks

- ▶ AngularJS
 - ▶ ReactJS
 - ▶ Knockout
 - ▶ Polymer
- 

Serviços Web (Web Services)

- ▶ Mistura entre Arquitetura Orientada a Eventos e outras Arquiteturas
 - ▶ Permite que todo o processamento de interface fique no lado do cliente
 - ▶ Permite que todo o armazenamento e regras de negócio fique no lado do servidor
- 