

Sistema Servidor Cliente



Educação a Distância
Cruzeiro do Sul Educacional
Campus Virtual

Material Teórico



Tecnologias Cliente / Servidor na Internet

Responsável pelo Conteúdo:

Prof. Esp. Marcio Funes

Revisão Técnica:

Prof. Ms. Luiz Carlos Reis

Revisão Textual:

Profa. Ms. Claudio Brites



- Introdução
- ASP
- PHP
- .NET



Objetivo de APRENDIZADO

- Nesta unidade, estudaremos as tecnologias envolvidas no desenvolvimento de soluções cliente/servidor na internet – conhecer quais delas estão disponíveis no mercado é fundamental para qualquer profissional que deseja seguir carreira na área de Web.

A cada dia, novos conceitos, novas ferramentas e novas linguagens surgem no mercado voltado a soluções web. Isso se dá porque diversas empresas querem, ao oferecerem serviços, beneficiarem-se da praticidade e velocidade com que seus clientes podem acessar tais serviços on-line. Devido a essa grande demanda, cada vez mais profissionais são requisitados com propostas de salários atrativas e diversos benefícios.

Peço, inicialmente, que você leia o material disponibilizado para, depois, realizar os exercícios propostos. Essa prática vai auxiliá-lo a conhecer o conteúdo antes de partir para a aplicação da aprendizagem. Temos vários tipos de atividades nesta unidade, que deverão ser realizadas como parte obrigatória do conteúdo, para o domínio do conhecimento adquirido.

É importante também que você assista à apresentação em PowerPoint narrado desta unidade, pois ela sintetiza o conteúdo e será de grande auxílio para consolidação do que for visto no material teórico.

É fundamental a sua participação no debate com seus colegas sobre os temas propostos; afinal, os que trazem experiências poderão compartilhá-las com os que não a possuem ou que estão em início de carreira.

Contextualização

Em 1995, Rasmus Lerdorf, um desenvolvedor canadense-dinamarquês, queria implementar funcionalidades de consulta de estatísticas baseadas em números de acesso a currículo on-line. Para isso, criou uma série de scripts no formato CGI na linguagem C, batizando-os de Personal Home Page Tools. Era o início da linguagem que conhecemos hoje como PHP, que está presente em diversas e diversas páginas da Internet, permitindo funcionalidades inúmeras a todas elas.

Assim como o PHP, outras soluções que tragam novas possibilidades a sistemas cliente/servidor na internet são muito procuradas por empresas que querem seus produtos e serviços acessíveis de um modo fácil, rápido e confiável na web, criando um nicho de mercado enorme para empresas de tecnologia e desenvolvedores.

Seguindo essa tendência de busca por novas soluções, em 1989, um holandês chamado Guido van Rossum decidiu criar uma linguagem diferente de tudo o que já se tinha visto. Ele queria uma linguagem que fosse aberta e comunitária, e que trouxesse ao desenvolvedor novas possibilidades. Em 1991, Guido lançou essa linguagem, chamada Python, que tem ganhado cada vez mais adeptos – principalmente quando temos uma solução Python aliada ao framework Django, muito procurada por desenvolvedores web que desejam criar soluções cliente/servidor em web.

Independentemente da linguagem ou solução escolhida, o papel de novas tecnologias aplicadas a esses contextos é muito importante e cresce cada vez mais, ampliando a demanda de novos profissionais. Portanto, seja bem-vindo, ou bem-vinda, a esse novo cenário de soluções, escolha uma tecnologia que mais se encaixa em seu perfil e siga em frente, pois a nossa área necessita de seus talentos.

Introdução



Ao fazermos uma rápida busca pela internet a procura de tecnologias e linguagens que podemos utilizar no contexto de Web, perceberemos a quantidade e diversidade de soluções que o mercado oferece aos profissionais de tecnologia. Dentre essas soluções, algumas se destacam e é muito importante que você as conheça e saiba suas aplicabilidades e exemplos.

Nesse capítulo, você encontrará algumas dessas tecnologias, que podemos utilizar na internet para criar possibilidades em sistemas cliente/servidor. Como cada tecnologia e linguagem possuem muitas informações, modos de utilização, declaração de variáveis e detalhes particulares, abaixo você encontra o essencial de cada uma dessas tecnologias para auxiliar na busca e formação de seu conhecimento.

ASP



Em 1996, durante a *Site Builders Conference and the Professional Developers Conference*, a Microsoft apresentou ao mundo sua nova solução para ambientes de programação focados em servidores web: o ASP – abreviação de *Active Server Pages* ou, em tradução livre, *Páginas de Servidor Ativas*. Ele possui a função de criar páginas dinâmicas, interativas e possibilitar novos recursos aos clientes que já utilizavam soluções Microsoft aplicadas à Internet (LOTAR, 2010).

O ASP foi criado exclusivamente para servidores Microsoft que utilizavam o IIS (*Internet Information Service*) para hospedar páginas e prover acesso a elas – por isso era exclusivo para desenvolvedores e empresas que já utilizavam soluções Microsoft. O IIS permitia hospedar páginas em HTML e, como complemento, permitia incluir scripts ASP nas funcionalidades das páginas hospedadas.

Primeiros passos em ASP

Ao abrir uma página com extensão .asp, podemos observar que ela irá possuir códigos em HTML e scripts contendo instruções em ASP, que permitem o dinamismo das páginas e a interação com os usuários. Outro fato importante é que os scripts ASP não são executados diretamente na máquina do cliente, cabe ao servidor que hospeda a página processar as instruções e entregar a seu cliente a funcionalidade, esse fator permite que diversos navegadores possam acessar páginas ASP sem problemas.

Vejamos agora um exemplo simples de aplicação de um script ASP em uma página HTML, trazendo assim algum dinamismo a essa página. Um fator muito importante é que os scripts ASP não podem ser visualizados pelo cliente, pois ficam apenas implementados no servidor, deixando o cliente apenas visualizar o código HTML puro.

Abaixo, temos um código simples em HTML:

```
<HTML>
  <HEAD>
    <TITLE> Minha página em ASP </TITLE>
  </HEAD>
  <BODY>
    A hora atual é 19:24:31 e estamos no dia 14
  </BODY>
</HTML>
```

Perceba que a página deseja mostrar ao cliente a hora atual e o dia, porém, essa função não é possível apenas com o código HTML. Vamos implementar um script ASP para isso:

```
<HTML>
  <HEAD>
    <TITLE> Minha página em ASP </TITLE>
  </HEAD>
  <BODY>
    A hora atual é <%=time%> e estamos no dia <%=day(now)%>
  </BODY>
</HTML>
```

Vejamos um exemplo mais elaborado, no qual temos uma página simples em HTML solicitando informações a um cliente e, logo após, uma página ASP que exibe os valores recebidos.

Considere o formulário HTML abaixo, solicitando o Nome e o Telefone do cliente:

The image shows a screenshot of a web page with a form. It has three main elements: a text input field labeled "Nome" with a placeholder box, a text input field labeled "Telefone:" with a placeholder box, and a blue rectangular button labeled "Enviar formulário".

Figura 1 – Exemplo de Formulário

Vejamos abaixo o código HTML utilizado para criar o formulário acima:

- **Meusdados.html**

```
<HTML>
  <HEAD>
    <TITLE>Meus Dados</TITLE>
  </HEAD>
  <BODY>
    <FORM name="Form" action="request.asp" method="get">
      Nome
      <INPUT type="text" name="NomeCompleto" size="30"><p>
      Telefone:
      <INPUT type="text" name="Telefone" size="15"><p>
      <INPUT type="submit" value="Enviar formulário">
    </FORM>
  </HTML>
```

Veja no código que a tag `<Form>` possui o `action="request.asp"`, enviando os dados pelo método `get` (`method="get"`) para a página `request.asp`.

Vejamos agora como deverá ser o código da página `resquest.asp` e como utilizar as informações da página `Meusdados.html`

```
<HTML>
  <HEAD><TITLE>Minha página ASP</TITLE>
  <%
    NomeCompleto = request.querystring("NomeCompleto")
    Telefone = request.querystring("Telefone")
  %>
  </HEAD>
  <BODY>
    Seus Dados:
    Nome: <% response.write(NomeCompleto)%><BR>
    Telefone: <% response.write(Telefone)%></P>
  </BODY>
</HTML>
```

Veja que conseguimos a informação digitada no formulário da página `Meusdados.html` através do `resquest.querystring`. Perceba também que utilizamos o mesmo nome dado no tag `<INPUT> NomeCompleto`. Conseguimos, assim, exibir para o cliente a informação através do `<% response.write(NomeCompleto)%>`.

ASP.NET

Como toda tecnologia sempre está em constante mudança e melhoramento, não seria diferente com o ASP. O ASP.NET é considerado a próxima geração do ASP, e é muito utilizado em sites comerciais. Como principal vantagem, permite orientação a objetos e pode ser implementado no Visual Studio .NET.



Acesse: <http://www.asp.net/get-started>

Esse é o site oficial da Microsoft, que permite fazer download do Visual Studio e começar a desenvolver soluções ASP.NET – acesso obrigatório para programadores!

PHP



Responsável por grande parte da evolução tecnológica voltada a desenvolvimento web, o PHP é uma linguagem para páginas dinâmicas e possibilita muitas funções em uma ambiente Cliente/Servidor. Uma grande vantagem do PHP é seu código ser livre e gratuito, possibilitando a qualquer um utilizar sua estrutura sem preocupações de diretos. Além de gratuito, o PHP possui alto nível de segurança e é totalmente compatível com o Linux, principalmente em soluções web do tipo LAMP: Linux + Apache + MySQL + PHP.

Estrutura da Linguagem

O PHP pode ser utilizado em e-commerce, páginas pessoais, sistemas de intranet e em gerenciamento de banco de dados. Assim como outras soluções, a estrutura da linguagem se inicia com a requisição de página do cliente, o servidor por sua vez procura a página solicitada e interpreta o código PHP contido nela, retornando o resultado da função ali contida (MILANI, 2010).

Vejamos um exemplo de uma página PHP:

```
<HTML>
<HEAD>
    <TITLE>Insira seus Dados</TITLE>
</HEAD>
<BODY>
    <?
        echo "Esse script é PHP";
    ?>
</BODY>
<?HTML>
```

Perceba que assim como em outras linguagens, também utilizamos o HTML como linguagem base, codificando o PHP entre suas tags. Para diferenciar o PHP, utilizamos o <?.

Vejamos outro exemplo:

```
<HTML><BODY>
<?
$texto1 = "PHP";
$texto2 = "Esse é um exemplo simples de $texto1";
$echo $texto2;
?>
</BODY></HTML>
```

O código acima exibiria no navegador a seguinte frase:

Esse é um exemplo simples de PHP.

Perceba que foi criado uma variável chamada \$texto1, que armazenou o valor “PHP”, e outra chamada \$texto2, que armazenou a frase. Podemos exibir no navegador a frase completa através do \$echo.

Vejamos outro exemplo, agora utilizando alguma funcionalidade:

```
<HTML>
<BODY>
<?
function soma($valor1, $valor2){
    return ($valor1 + $valor2)/2;;
}
$media = soma(10, 20);
echo "A media é $media";
?>
</BODY>
</HTML>
```

O código acima possui a função de calcular uma simples média, veja que foi criada uma função chamada soma através do function, e reservada duas variáveis: \$valor1 e \$valor2. Essa função retorna essas duas variáveis, sendo divididas por 2, caracterizando uma média. Logo após, temos outra variável chamada \$media, recebendo o retorno da função media com os valores das variáveis definidos como 10 e 20. Para finalizar, exibimos no navegador através do echo o resultado da média.

A linguagem permite utilizar funções como de IF:

```
<?
$num1 = 10;
IF ($num == 10){
echo "Essa linha será exibida";
}
?>
```

E também funções de repetição do o Do WHILE:

```
?>
$var = 10;
do{
echo "Repetindo";
}while ($var < 10);
?>
```

Acima, pudemos ver alguns exemplos de como é estruturada a linguagem PHP, esperamos que tenha se sentido incentivado a buscar conhecimento sobre essa linguagem. O PHP não se resume a essas funções, portanto, se você estava a procura de uma linguagem web que possibilita criar soluções Cliente/Servidor, acima estão os primeiros passos.

Para que você se aprofunde no PHP, sugerimos visitar os sites abaixo:

- » Site oficial do PHP para downloads: <http://www.php.net/>
- » Manual que descreve as funções do PHP: http://www.php.net/manual/pt_BR/
- » No site do W3schools temos diversos tutoriais: <http://www.w3schools.com/PHP/>
- » Grupo dos Desenvolvedores de PHP do Estado de São Paulo: <http://phpsp.org.br/>



Ainda na temática Microsoft, temos a plataforma .NET, lançada na virada do milênio e que possui dois objetivos principais: o primeiro é facilitar o trabalho do desenvolvedor, tornando a construção do projeto prática e atendendo a qualquer necessidade do cliente; a segunda é o de se estabelecer como uma solução viável no ambiente de Web Services, tendo como principal concorrente as soluções Java.

.NET (lê-se dotNet) é uma tecnologia focada em atender os padrões que estão incorporados na internet porém sem deixar de lado as características que são estabelecidas pela Microsoft, o desenvolvedor pode então se beneficiar com a facilidades de desenvolvimento presentes no produtos Microsoft sem se fechar aos padrões da internet e assim comunicar com outras tecnologias. (LOTAR 2010)

.NET Framework

Para que você possa construir seus projetos em .NET, a Microsoft oferece o .NET Framework como um ambiente de programação para aplicações Web e Web Service de modo simplificado. Ele suporta linguagens como C++, C#, Visual Basic, ASP.NET, dentre outras.

Como pacote de programas principal voltado a .NET Framework, a Microsoft disponibiliza o Microsoft Visual Studio, que é atualizado constantemente, fornecendo meios para se desenvolver nas linguagens suportadas pela .NET Framework. A seguir, alguns exemplos de codificação nesse ambiente.

Vejamos abaixo um exemplo da linguagem C#:

```
using System;
class HelloWorld
{
    static void Main()
    {
        Console.WriteLine("Hello World");
    }
}
```

Veja que o programa começa com o using System, no qual indicamos que o programa abaixo pede as classes e funções presentes no System. Logo após, temos a classe HelloWorld e, como toda classe, com ela podemos criar objetos e com eles atributos e métodos. Porém, nessa classe, apenas utilizamos a classe Console, possível de ser utilizada devido a termos usado o System no começo do código e utilizar o WriteLine, que está presente dentro da classe Console – podemos ver essa relação pelo “.” entre o Console e o WriteLine para, assim, esse código mostrar ao usuário a frase Hello World.

Vejamos outro exemplo:

```
using System;
using System.Collections.Generic;
using System.Text;

Public class Cliente{

    private long      id;
    private double    renda;
    private DateTime  nascimento;
    private string    nome;

    Public Cliente(long id, double renda, DateTime nascimento, string nome){

        this.id = id;
        this.renda = renda;
        this.nascimento = nascimento;
        this.nome = nome;
    }
}
```

Para que você conheça mais sobre o .Net, sugiro os sites abaixo:

- » Revista .Net: <http://www.devmedia.com.br/revista-easy-dotnet-magazine>
- » Conteúdo Extra: <http://www.devmedia.com.br/dotnet/>
- » Site oficial da Microsoft .Net: <http://www.microsoft.com/net>

ColdFusion

Ao começar a estudar essa tecnologia, você perceberá que ColdFusion não se resume apenas a uma linguagem, mas do que isso, ela fornece soluções web para desenvolvimento de sites, portais, intranets e demais aplicações web. Como ela possui funções para auxiliar em sistemas Cliente/Servidor, é fundamental conhecê-la.

Seu funcionamento

Inicialmente chamado de *Cold Fusion*, foi criado em 1995 pelos irmãos Allaire. Logo depois foi comprado pela empresa Adobe. Atualmente, para se trabalhar com essa tecnologia, primeiro temos que entender que CF (*ColdFusion*) também é um servidor de aplicações web, ou seja, o *ColdFusion* é um ambiente de desenvolvimento de aplicações web dinâmicas com suporte à linguagem CFML – abreviação de *ColdFusion Markup Language*. Nesse ambiente, podemos utilizar banco de dados, recursos de e-mail e aplicações Java, o que torna o CF atraente a diversos desenvolvedores (BRADLEY, 2000).

Você pode desenvolver páginas utilizando a linguagem CFML, que é baseada em tags, assim como o HTML, porém irá necessitar configurar um servidor CF para interpretar a linguagem. Primeiramente, o servidor realiza a leitura das páginas à procura de tags, que são iniciadas por CF, além de ler variáveis ou funções que contenham o sinal #.

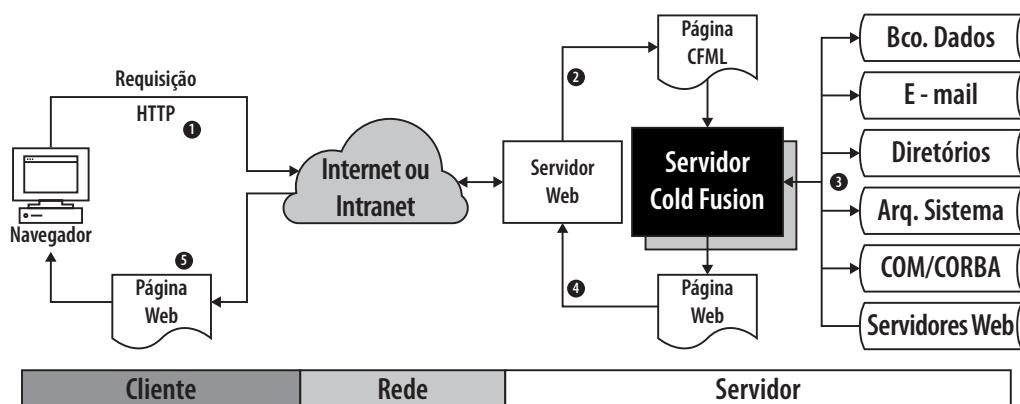


Figura 2 – Esquema ColdFusion

1. Quando o usuário faz alguma ação em um site, o navegador envia uma requisição HTTP para o servidor Web;
2. O servidor Web, recebendo a requisição, envia os dados para um servidor ColdFusion através de API;

3. O servidor de ColdFusion, por sua vez, interpreta os códigos de CFML que estão programadas dentro das páginas. Dependendo da programação CFML da página, o servidor então faz interações com servidores de e-mail, banco de dados, diretórios e demais serviços;
4. Após processar a programação CFML e acessar os serviços que ali estão implementados, o servidor ColdFusion gera dinamicamente uma página HTML que será enviada ao servidor Web;
5. Por último, o servidor Web retorna a página HTML para o usuário.

Veja um exemplo de CFML em uma página HTML:

```
<HTML>
<HEAD>
<TITLE>Olá Mundo</TITLE>
</HEAD>
<BODY>
<CFSET OLA='Olá Mundo'>
<CFOUTPUT>
<FONT SIZE="10">#ola#</FONT>
</CFOUTPUT>
</BODY>
</HTML>
```

Veja outro exemplo, utilizando acesso a banco de dados, principalmente em aplicações no contexto de Cliente/Servidor.

```
<CFQUERY DATASOURCE="AgendaContatos" NAME="Agenda">
SELECT * FROM Contatos
</CFQUERY>

<HTML>
<HEAD><TITLE>Teste Banco</TITLE></HEAD>
<BODY>
<H1> Testando Agenda </H1>
<CFOUTPUT QUERY="Agenda">
#home#.#telefone#
<CFOUTPUT>
</BODY>
</HTML>
```

Iremos agora conhecer uma das tecnologias que vem ganhando cada vez mais espaço no mercado. Em 1991, um holandês chamado Guido Van Rossum lança ao mundo a linguagem Python em busca de novas possibilidades na programação interpretada. De código aberto e disponível para vários sistemas operacionais, Python tem a característica de ser interpretada pois não necessita ser compilada, ou seja, traduzida para a linguagem de máquina; ela apenas precisa ser “lida” por um interpretador (assim como HTML), que irá traduzir as instruções que ali estão presentes (BROOKS-BILSON, 2003).

O nome Python foi dado em homenagem ao programa de TV britânico chamado Monty Python's Flying Circus, e não em associação a cobra Python. Além de interpretação, Python não necessita que declaremos variáveis como pré-antecedente de programação; a declaração de variáveis ocorre de forma dinâmica e bem intuitiva ao programador. Por ser uma linguagem de alto nível, possui recursos com strings, listas, dicionários, classes e, como característica fundamental, Python é orientado a objetos (LABAKI, 2010).

Vejamos agora alguns exemplos de códigos em Python:

Começando com o simples Hello Word, podemos fazer da seguinte forma:

```
a = "Hello"  
b = "World"  
print a, b
```

Como Saída temos: Hello World

Em Python, não é necessário dizermos que a variável a é do tipo Char. Ao receber um valor, a definição do tipo de variável é dinâmica.

Veja outra opção quando se trata de listas, vamos às declarações:

```
numeros = [1, 2, 3]  
opcoes = ["sim", "não"]  
listas = [números, opções]
```

Em nenhum momento foi necessário preparar a variável para receber vários valores em uma lista. Ao exibir essas listas, temos como saída:

```
print numeros[]  
Saída: 1, 2, 3
```

Se quiser que seja exibido apenas o 3º número, podemos executar a seguinte saída:

```
print numeros[2]  
Saída: 3 // entendendo que a contagem começa a partir do 0;
```

Ou ainda para exibir as duas listas podemos apenas gerar essa saída:

```
print listas  
Saída: 1, 2, 3, sim, não
```

Ou ainda, podemos ver uma aplicação mais elaborada, uma rotina muito conhecida por estudantes de programação, descobrindo números primos, veja como podemos programar essa rotina facilmente no Python:

```
p = input("Entre com um número natural: ")  
primo = True  
for i in range(2, n):
```

```
if n%i == 0:  
    primo = False  
if primo:  
    print n, "numero primo"  
else:  
    print n, "não é numero primo"
```

Com os exemplos acima esperamos que tenha percebido que a linguagem Python é extremamente simples e intuitiva, e por isso muito indicada para quem está dando seus primeiros passos na programação. Veja na seção Material Complementar onde você pode encontrar mais conteúdos de Python.

Material Complementar

Como complemento desta unidade, sugiro a leitura completa do capítulo I do livro:

» MENEZES, N N C. **Introdução a programação com Python.** São Paulo: Novatec, 2009.

Sugiro, também, a leitura completa do livro:

» LOTAR, A. **Como Programar com ASP.NET e C#.** Novatec, 2010.

Referências

- BROOKS-BILSON, R. **Programming ColdFusion MX: Creating Dynamic Web Applications** 2/E, O'Reilly, 2003
- LABAKI, J. **Introdução ao Python** – Módula A, Grupo Python, UNESP – Ilha Solteira
- HETLAND, M L. **Beginning Python**: from novice to Professional. Apress, 2005.
- LOTAR, A. **Como Programar com ASP.NET e C#**. Novatec, 2010.
- MILANI, André. **Construindo Aplicações Web com PHP e MySQL**. Novatec, 2010.
- BRADLEY, M Z. **ColdFusion – Guia rápido para Desenvolvimento na Web**. Ciência Moderna, 2000.

Anotações





Educação a Distância
Cruzeiro do Sul Educacional
Campus Virtual

www.cruzeirodosulvirtual.com.br
Campus Liberdade
Rua Galvão Bueno, 868
CEP 01506-000
São Paulo SP Brasil
Tel: (55 11) 3385-3000



Universidade
Cruzeiro do Sul



UNICID
Universidade
Cidade de S. Paulo



UNIFRAN
Universidade
de Franca



UDF
Centro
Universitário



Módulo
Centro
Universitário