

# ASP.NET MVC 5 - CONTROLADORES (A6LP2)

## 02 – CONTROLADORES

# AGENDA

- Controlador (Controller)
- Métodos de Ação (Action Methods) e Resultados da Ação (ActionResult)
- Seletores de Ação (Action Selectors)
- Envio de Dados do cliente para o Controlador

# ASP.NET MVC – CONTROLADOR

## Controlador

- namespace System.Web.Mvc
- Classe Base Controller

```
namespace PrimeiraAppWeb.Controllers
{
    public class HomeController : Controller
    {
        // GET: Home
        public ActionResult Index()
        {
            return View();
        }

        public ActionResult Contato()
        {
            return View();
        }
    }
}
```

# ASP.NET MVC – MÉTODOS DE AÇÃO

## Ação Padrão

- Index
- home/

```
namespace PrimeiraAppWeb.Controllers
{
    public class HomeController : Controller
    {
        // GET: Home
        public ActionResult Index()
        {
            return View();
        }

        public ActionResult Contato()
        {
            return View();
        }
    }
}
```

# ASP.NET MVC – RESULTADOS DA AÇÃO

## ActionResult

- Retorno de um método de ação

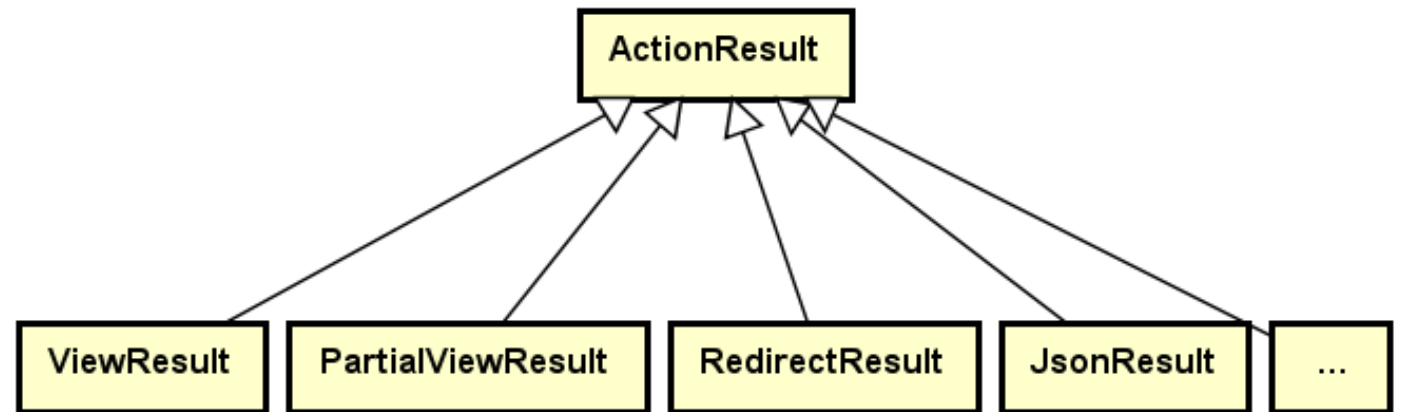
```
namespace PrimeiraAppWeb.Controllers
{
    public class HomeController : Controller
    {
        // GET: Home
        public ActionResult Index()
        {
            return View();
        }

        public ActionResult Contato()
        {
            return View();
        }
    }
}
```

# ASP.NET MVC – RESULTADOS DA AÇÃO

## ActionResult

- Retorno de um método de ação
- Classe base para todos os resultados de ação
- Vários tipos derivam de ActionResult



# ASP.NET MVC – RESULTADOS DA AÇÃO

## ViewResult

- Mais comum
- Renderiza uma view como pagina web html (incluindo o layout)
- Método auxiliar (Helper): View

```
public class HomeController : Controller
{
    public ViewResult Index()
    {
        return View();
    }
}
```

/Views/Shared/\_Layout.cshtml

/Views/Home/Index.cshtml

# ASP.NET MVC – RESULTADOS DA AÇÃO

## PartialViewResult

- Apenas a view sem o layout
- Método auxiliar (Helper):  
PartialView

```
public class HomeController : Controller
{
    public PartialViewResult Index()
    {
        return PartialView();
    }
}
```

/Views/Home/Index.cshtml



# ASP.NET MVC – RESULTADOS DA AÇÃO

## RedirectResult

- Redireciona para outro método de ação usando sua URL.
- Redirecionamento para fora
- Método auxiliar (Helper):  
Redirect

```
public RedirectResult Acao1()
{
    //Lógica da Ação 1

    return Redirect("http://www.google.com.com");
}
```

# ASP.NET MVC – RESULTADOS DA AÇÃO

## RedirectToRouteResult

- Redireciona para outro método de ação.
- Método auxiliar (Helper): RedirectToAction ou RedirectToRoute

```
public ViewResult Index()
{
    return View();
}

public RedirectToRouteResult Acao2()
{
    //Lógica da Ação 2

    return RedirectToAction("Index");
}
```

# ASP.NET MVC – RESULTADOS DA AÇÃO

## RedirectToRouteResult

- Redireciona para outro método de ação.
- Método auxiliar (Helper): RedirectToAction ou RedirectToRoute

```
public ViewResult Index()
{
    return View();
}

public RedirectToRouteResult Acao2()
{
    //Lógica da Ação 2

    return RedirectToAction("Index", "Home");
}
```

# ASP.NET MVC – RESULTADOS DA AÇÃO

## RedirectToRouteResult

- Redireciona para outro método de ação.
- Método auxiliar (Helper): RedirectToAction ou RedirectToRoute

```
public ActionResult Index()
{
    return View();
}

public RedirectToRouteResult Acao2()
{
    //Lógica da Ação 2

    return RedirectToAction("Index", "Home", new { id = 10});
}
```

# ASP.NET MVC – RESULTADOS DA AÇÃO

## ContentResult

- Retorna um tipo de conteúdo definido pelo usuário
- Método auxiliar (Helper): Content
- MIME Type (Tipo de Media)

```
public ContentResult Conteudo1()  
{  
    return Content("Olá Mundo!");  
}
```

# ASP.NET MVC – RESULTADOS DA AÇÃO

## ContentResult

- Retorna um tipo de conteúdo definido pelo usuário
- Método auxiliar (Helper): Content
- MIME Type (Tipo de Media)

```
public ContentResult Conteudo1()  
{  
    return Content("Olá Mundo!", "text/plain");  
}
```

# ASP.NET MVC – RESULTADOS DA AÇÃO

## ContentResult

```
public ContentResult Conteudo1()
{
    return Content("<msg><texto>Olá Mundo</texto></msg>", "text/xml");
}
```

- Retorna um tipo de `ContentResult` definido pelo usuário
- Método auxiliar (Helper): `Content`
- MIME Type (Tipo de Media)

## ASP.NET MVC – RESULTADOS DA AÇÃO

```
public ActionResult Conteudo1()  
{  
    return Content("{\"msg\":\"texto\": \"Ola mundo\"}", "application/json");  
}
```

- Retorna }  
definido pelo usuário
- Método auxiliar (Helper):  
Content
- MIME Type (Tipo de Media)



# ASP.NET MVC – RESULTADOS DA AÇÃO

## JsonResult

- Retorna um objeto Json Serializado
- Método auxiliar (Helper): Json

```
public JsonResult Dados()  
{  
    var dados = new {  
        Id = 1,  
        Nome = "João",  
        Situacao = true  
    };  
  
    return Json(dados);  
}
```

# ASP.NET MVC – RESULTADOS DA AÇÃO

## JsonResult

- Retorna um objeto Json Serializado
- Método auxiliar (Helper): Json
- Habilitar requisições GET

## Erro de Servidor no Aplicativo '/'.

*This request has been blocked because sensitive information could be disclosed to third party web sites when this is used in a GET request. To allow GET requests, set JsonRequestBehavior to AllowGet.*

**Descrição:** Ocorreu uma exceção sem tratamento durante a execução da atual solicitação da Web. Examine o rastreamento de pilha para obter mais informações sobre o erro e onde foi originado no código.

**Detalhes da Exceção:** System.InvalidOperationException: This request has been blocked because sensitive information could be disclosed to third party web sites when this is used in a GET request. To allow GET requests, set JsonRequestBehavior to AllowGet.

### Erro de Origem:

Exceção sem tratamento foi gerada durante a execução da atual solicitação da Web. As informações relacionadas à origem e ao local da exceção podem ser identificadas usando-se o rastreamento de pilha de exceção abaixo.

### Rastreamento da Pilha:

# ASP.NET MVC – RESULTADOS DA AÇÃO

## JsonResult

- Retorna um objeto Json Serializado
- Método auxiliar (Helper): Json
- Habilitar requisições GET

```
public JsonResult Dados()
{
    var dados = new {
        Id = 1,
        Nome = "João",
        Situacao = true
    };

    return Json(dados, JsonRequestBehavior.AllowGet);
}
```

# ASP.NET MVC – RESULTADOS DA AÇÃO

## HttpStatusCodeResult

- Retorna um código de resposta HTTP específico e uma descrição.
- Método auxiliar (Helper): Nenhum

```
public HttpStatusCodeResult Status()  
{  
    return new HttpStatusCodeResult(404);  
}
```

# ASP.NET MVC – RESULTADOS DA AÇÃO

## HttpStatusCodeResult

- Retorna um código de resposta HTTP específico e uma descrição.
- Método auxiliar (Helper): Nenhum

```
public HttpStatusCodeResult Status()  
{  
    return new HttpStatusCodeResult(404);  
}
```

HTTP code	Meaning
200	OK
4xx	Bad request (client's fault)
5xx	Failed request (server's fault)
401	Unauthorized request
404	Resource not found
500	Internal error (bug)
503	Server overloaded

# ASP.NET MVC – RESULTADOS DA AÇÃO

## FilePathResult

- Envia o conteúdo de um arquivo à resposta.
- Método auxiliar (Helper): File

```
public FilePathResult VerArquivo()
{
    var appData = Server.MapPath("~/App_Data");
    var path = Path.Combine(appData, "teste.pdf");
    return File(path, "application/pdf");
}
```

# ASP.NET MVC – RESULTADOS DA AÇÃO

## FilePathResult

- Envia o conteúdo de um arquivo à resposta.
- Método auxiliar (Helper): File
- FileStreamResult

```
public FilePathResult VerArquivo()
{
    var appData = Server.MapPath("~/App_Data");
    var path = Path.Combine(appData, "teste.pdf");
    return File(path, "application/pdf", "teste.pdf");
}
```

# ASP.NET MVC – SELETORES DE AÇÃO

- Atributos aplicados as Actions
  - ActionName
  - NonAction
  - ActionVerbs



# ASP.NET MVC – SELETORES DE AÇÃO

## ActionName

- Define um nome diferente para uma ação
- É possível ter caracteres não permitidos como identificador
- Combinação com Action Verbs e definir o mesmo nome para duas Ações

```
[ActionName("cadastrar-produto")]  
public ActionResult CadastrarProduto()  
{  
    return Content("Cadastrar Produto");  
}
```

```
[ActionName("cadastrar-produto")]  
[HttpPost]  
public ActionResult SalvarProduto()  
{  
    return Content("Salvar Produto");  
}
```

# ASP.NET MVC – SELETORES DE AÇÃO

## NonAction

- Indica que um método não é uma Ação
- Não é possível acessá-lo via requisições HTTP

```
[NonAction]
public string AlgumaAcao()
{
    //Lógica interna ..

    return "";
}
```

# ASP.NET MVC – SELETORES DE AÇÃO

## ActionVerbs

- Get
- Post
- Options
- Head
- Put
- Delete
- Patch

# ASP.NET MVC – SELETORES DE AÇÃO

## [HttpGet]

- Obter um recurso do servidor
- Parâmetros podem ser definido via query string

```
[HttpGet]  
public ActionResult TesteGet()  
{  
    return Content("Teste Get");  
}
```

# ASP.NET MVC – SELETORES DE AÇÃO

## [HttpPost]

- Criar um novo recurso
- Valores podem ser definidos no corpo da requisição

```
[HttpPost]
public ActionResult TestePost()
{
    return Content("Teste Post");
}
```

# ASP.NET MVC – SELETORES DE AÇÃO

## [HttpOptions]

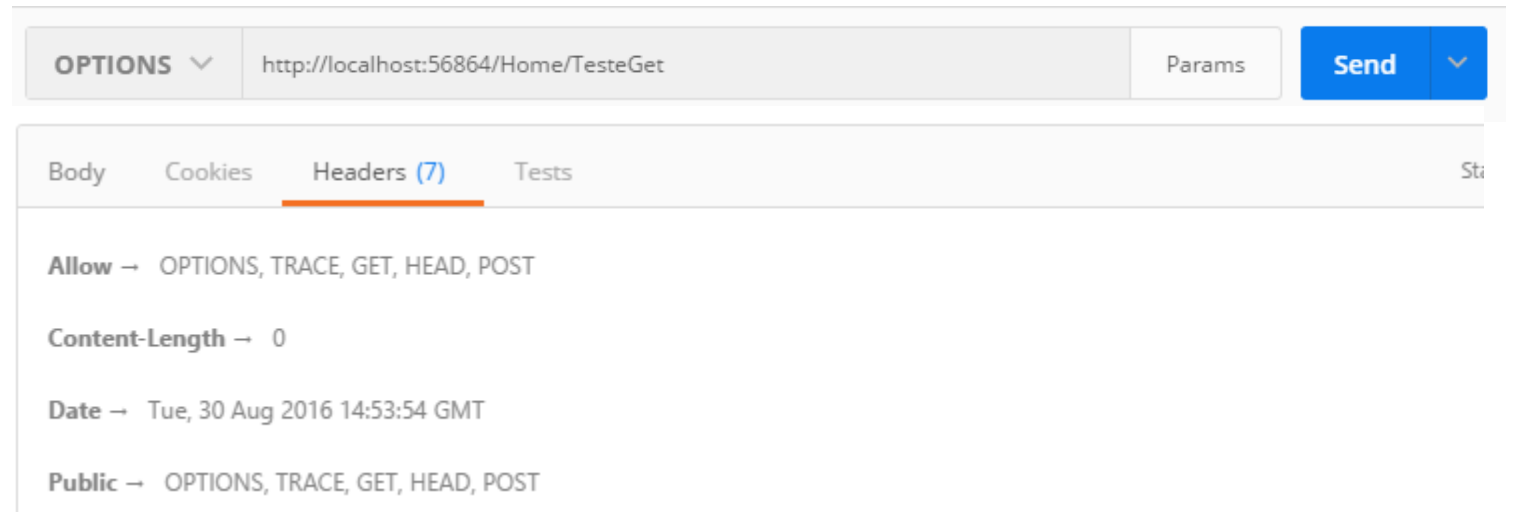
- Requisição de informações sobre opções de comunicação com o servidor

```
[HttpOptions]
public ActionResult TesteOptions()
{
    return Content("Teste Options");
}
```

# ASP.NET MVC – SELETORES DE AÇÃO

## [HttpOptions]

- Requisição de informações sobre opções de comunicação com o servidor



# ASP.NET MVC – SELETORES DE AÇÃO

## [HttpHead]

- Idêntico ao GET mas o servidor não devolve o corpo da mensagem
- Acesso as informações do header sem precisar carregar o conteúdo todo

## [HttpHead]

```
public ActionResult TesteHead()  
{  
    return Content("Teste Head");  
}
```



HEAD

http://localhost:56864/Home/TesteGet

Params

Send

Save

key

value

Bulk Edit

Authorization

Headers

Body

Pre-request Script

Tests

Generate Code

key

value

Bulk Edit

Presets

Body

Cookies

Headers (8)

Tests

Status: 200 OK

Time: 25 ms

Cache-Control → private

Content-Encoding → gzip

Content-Length → 127

Content-Type → text/html; charset=utf-8

Date → Tue, 30 Aug 2016 14:56:19 GMT

Vary → Accept-Encoding

X-Powered-By → ASP.NET

HEAD ▾

http://localhost:56864/Home/TesteGet

Params

Send ▾

Save ▾

key

value

Bulk Edit

Authorization

Headers

Body

Pre-request Script

Tests

Generate Code

key

value

Bulk Edit

Presets ▾

Body

Cookies

Headers (8)

Tests

Status: 200 OK



Time: 25 ms

Pretty

Raw

Preview

HTML ▾



1

# ASP.NET MVC – SELETORES DE AÇÃO

[HttpPut] [HttpPatch] [HttpDelete]

- Desabilitados por padrão no ASP.NET MVC

## Removendo Headers

Cache-Control → private

Content-Length → 5041

Content-Type → text/html; charset=utf-8

Date → Tue, 30 Aug 2016 13:19:53 GMT

Server → Microsoft-IIS/10.0



X-AspNet-Version → 4.0.30319



X-Powered-By → ASP.NET



X-SourceFiles → =?UTF-8?B?

QzpcVXNlcnNcRG9taW5nb3NMdWNhc1xEZXNrdG9wXEIGU1AtU1BcTG

=

## Removendo Headers

Global.asax.cs

```
public class MvcApplication : System.Web.HttpApplication
{
    protected void Application_Start()
    {
        MvcHandler.DisableMvcResponseHeader = true;
        AreaRegistration.RegisterAllAreas();
        FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);
        RouteConfig.RegisterRoutes(RouteTable.Routes);
        BundleConfig.RegisterBundles(BundleTable.Bundles);
    }

    protected void Application_PreSendRequestHeaders(object sender, EventArgs e)
    {
        HttpContext.Current.Response.Headers.Remove("Server");
        HttpContext.Current.Response.Headers.Remove("X-AspNet-Version");
        HttpContext.Current.Response.Headers.Remove("X-AspNetMvc-Version");
        HttpContext.Current.Response.Headers.Remove("X-Powered-By");
    }
}
```

## Removendo Headers

Web.config

```
</appSettings>  
<system.web>  
  <authentication mode="None" />  
  <compilation debug="true" targetFramework="4.5.2" />  
  <httpRuntime targetFramework="4.5.2" enableVersionHeader="false" />  
</system.web>  
<system.webServer>  
  <modules>
```

# ASP.NET MVC – CONTROLADOR RECEBENDO DADOS

- Objetos de Contexto: HttpContext, Request, Response, RouteData
- Parâmetros nas Actions (Métodos do Controlador)
- Data Model Binding

# ASP.NET MVC – OBJETO REQUEST

## Objeto Request

- Request.Form
- POST

```
public ActionResult Login()
{
    var email = Request.Form["email"];
    var senha = Request.Form["senha"];

    if(email.Equals("joao@gmail.com") && senha.Equals("123"))
        return Content("Autenticado");

    return Content("Não Autenticado");
}
```



# ASP.NET MVC – OBJETO REQUEST

## Objeto Request

- Request.QueryString
- GET
- /Produto/BuscarProduto?id=1

```
public ActionResult BuscarProduto()
{
    var id = Request.QueryString["id"];

    //Busca o produto com id

    return Content(id);
}
```

# ASP.NET MVC – PARÂMETROS NAS ACTIONS

## Parâmetros no Método

- Nome do parâmetro deve ser igual o nome da chave de algum dos objetos:

Request.Form

Request.QueryString

Request.Files

RouteData.Values

```
public ActionResult Login(string email, string senha)
{
    //var email = Request.Form["email"];
    //var senha = Request.Form["senha"];

    if(email.Equals("joao@gmail.com") && senha.Equals("123"))
        return Content("Autenticado");

    return Content("Não Autenticado");
}
```

# ASP.NET MVC – PARÂMETROS NAS ACTIONS

## Parâmetros no Método

- Nome deve ser igual a um elemento de algumas das coleções:

Request.Form

Request.QueryString

Request.Files

RouteData.Values

```
public ActionResult BuscarProduto(string id)
{
    //var id = Request.QueryString["id"];

    //Busca o produto com id

    return Content(id);
}
```

# ASP.NET MVC – PARÂMETROS NAS ACTIONS

## Data Model Binding

- Model com os atributos

```
public class Usuario
{
    public string Email { get; set; }
    public string Senha { get; set; }
}

public ActionResult Login(Usuario usuario)
{
    if (usuario.Email.Equals("joao@gmail.com") && usuario.Senha.Equals("123"))
        return Content("Autenticado");

    return Content("Não Autenticado");
}
```

## EXERCÍCIO 1

- Crie um projeto “MVCTeste” e defina o TesteController com as seguintes actions:
  - **TesteView:** Dever retornar um objeto ViewResult
  - **TestePartialView:** Deve retornar um PartialViewResult
  - **Acao1:** Deve retornar para o cliente a mensagem “Ação 1” (MIME Type deve ser text/plain)
  - **Acao2:** Deve retornar para o cliente um xml simples (MIME Type deve ser application/xml)
  - **Acao3:** Deve retornar para o cliente um json simples (MIME Type deve ser application/json)

Faça os testes utilizando o



## EXERCÍCIO 2

- Crie a classe *Mensagem* na pasta *Model*:

<b>Mensagem</b>
Nome: String Email: String Texto: String

## EXERCÍCIO 3

- Crie o controlador Mensagem com as seguintes actions:
  - **Enviar:** Deve retornar para o cliente a mensagem “Formulário de Mensagem” (MIME Type deve ser text/plain)
  - **Enviar(Mensagem msg):** Deve ser acessível apenas pelo verbo POST e retornar um json do objeto mensagem

Faça os testes utilizando o



## EXERCÍCIO 4

- No controlador Mensagem crie a action **VerArquivo**. Esse método deve retornar um arquivo em formato pdf para o cliente.

Faça os testes utilizando o  **POSTMAN**



## EXERCÍCIO 5

- No controlador Home crie as seguintes actions:
  - **Sucesso:** deve retornar um objeto `HttpStatusCodeResult`
  - **NaoAutorizado:** deve retornar um objeto `HttpStatusCodeResult`
  - **NaoEncontrado:** deve retornar um objeto `HttpStatusCodeResult`
  - **ErroInterno:** deve retornar um objeto `HttpStatusCodeResult`

Faça os testes utilizando o



## EXERCÍCIO 6

- No controlador TesteVerbosHTTP crie as seguintes actions:
  - **TesteGet:** deve possuir o ActionVerb [HttpGet] e retornar um objeto ViewResult.
  - **TestePost:** deve possuir o ActionVerb [HttpPost], receber como parâmetro um objeto Produto e retornar um JsonResult desse objeto.
  - **TesteHead:** : deve possuir o ActionVerb [HttpHead] e retornar um objeto ContentResult.
  - **TesteOptions:** deve possuir o ActionVerb [HttpOptions] e retornar um objeto ContentResult.

Faça os testes utilizando o



Produto
Codigo: int Desc: string Valor: decimal

## EXERCÍCIO 7

- No controlador Home crie a action TesteArquivo:

```
[HttpPost]
public ActionResult TesteArquivo()
{
    HttpPostedFileBase arquivo = Request.Files["arquivo"];
    arquivo.SaveAs(Path.Combine(
        Server.MapPath("~/App_Data/"),
        Path.GetFileName(arquivo.FileName)
    ));
    return Content("");
}
```

Faça os testes utilizando o

