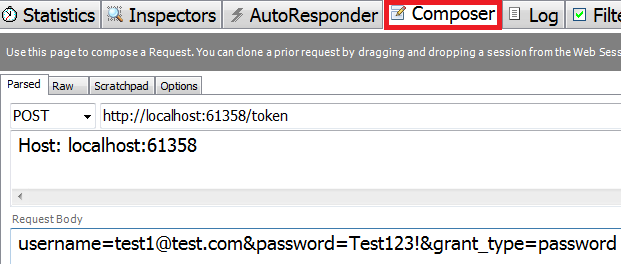
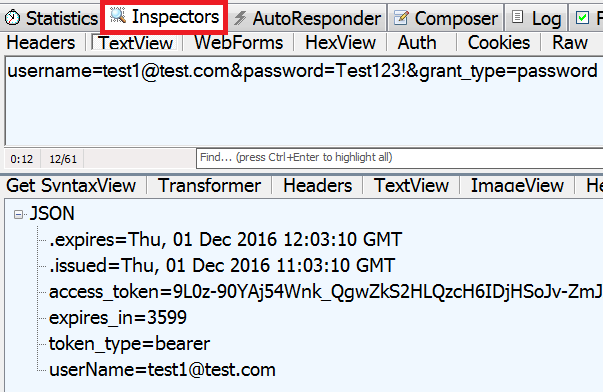
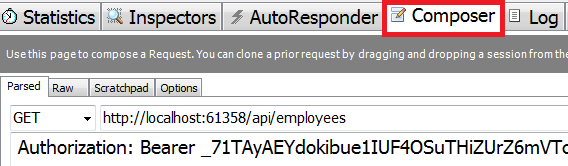
In this video we will discuss **how to test ASP.NET Web API token based authenticationthe using fiddler**. This is continuation to [Part 22](http://csharp-video-tutorials.blogspot.com/2016/12/using-aspnet-identity-with-web-api.html). Please watch [Part 22](http://csharp-video-tutorials.blogspot.com/2016/12/using-aspnet-identity-with-web-api.html)from [ASP.NET Web API tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhW7yoUOGNOfHurUE6bpOO2b) before proceeding.   
  
In our previous video we have registered a new user with the following email address and password. The username is also the email address.  
Email : test1@test.com  
Password : Test123!  
  
  
  
Now let's use fiddler and generate the access token using the above username and password. Use the Composer tab in Fiddler to compose a request. Fiddler configuration is shown below.

* Issue a POST request to /token
* In the request body include username and the password.
* We also need to set grant\_type=password. This indicates that we are presenting password for acquiring access token.

  
  
  
  
With the above configuration in place, click the **Execute**button in Fiddler. Notice we get the access token back. You can also see when the token is issued and when it expires.   
  
  
**Now let's understand how the access token is generated.**  
The code that generates the access token is provided by ASP.NET Web API out of the box. To see this code open the file "Startup.Auth.cs" that is present in App\_Start folder. Notice in the ConfigureAuth() method

* An instance of OAuthAuthorizationServerOptions is created
* The /Token end point to which we have posted username and password is specified in here
* The token expiry is specified using AccessTokenExpireTimeSpan property. In this case the token expires 14 days after it is issued. You can change this to meet your application needs.
* The Provider property is initialised with a new instance of ApplicationOAuthProvider class. This class has GrantResourceOwnerCredentials() method which verifies if the provided username and password are valid. If valid an access token is issued. The token is generated when context.Validated(ticket) method is called.

Now let us see how to call EmployeesController and retrieve employees data.  
  
If we issue a GET request to http://localhost:61358/api/employees we get 401 Unauthorized error. Since the EmployeesController is decorated with [Authorize]attribute, the request needs to be authenticated. So with every request we have to send the Bearer token using Authorization header as shown in fiddler below.   
   
  
In our next video we will discuss implementing the Login page for the sample application that we have been working with in this series.