MyDomainLoginDiscoveryHandler Interface

The handler used to implement the My Domain Login Discovery page, which is an intervi
(two-step) login process. First the user is prompted for a unique identifier such as an ema
or phone number. Then the handler determines (discovers) how to authenticate the user
the user enters a password or is directed to an identity provider's login page.

Namespace

Auth

Usage

Implement `MyDomainLoginDiscoveryHandler` to let My Domain users log in with something
than their username and password. This handler contains the logic to look up the user ba
the identifier value entered on the login page. The `Auth.MyDomainLoginDiscoveryHandler.l`
method is invoked when the identifier page is submitted and finds the user that correspo
submitted identifier. The `Auth.SessionManagement.finishLoginDiscovery` method sends th
the authentication mechanism and then logs in the user.

Register the handler from the My Domain Setup page. Under Authentication Configurati
the **Discovery** Login Page Type. For Login Discovery Handler, select this handler from the
Apex classes.

For an example, see MyDomainLoginDiscoveryHandler Example Implementation. For mo
search for My Domain Login Discovery in *Salesforce Help*.

- **MyDomainLoginDiscoveryHandler Method**
- **MyDomainLoginDiscoveryHandler Example Implementation**

MyDomainLoginDiscoveryHandler Method

`MyDomainLoginDiscoveryHandler` has the following method.

- **login(identifier, startUrl, requestAttributes)**
  Log in a Salesforce user given the specified identifier, such as email or phone num
  successful, redirect the user to the page specified by the start URL.

login(identifier, startUrl, requestAttributes)

Log in a Salesforce user given the specified identifier, such as email or phone number. If
redirect the user to the page specified by the start URL.

Signature

```
public System.PageReference login(String identifier, String startUrl, Map<String,Stri
requestAttributes)
```

Parameters

identifier

startUrl

Type: String

The page users see after successfully logging in to the My Domain subdomain.

requestAttributes

Type: Map <String, String>

Information about the login request based on the user's browser state when accessing page. `requestAttributes` passes in the MyDomainUrl, IpAddress, UserAgent, Platform, Application, City, Country, and Subdivision values. The City, Country, and Subdivision va from IP address geolocation.

Return Value

Type: System.PageReference

The URL of the page where the user is redirected to complete authentication.

Example

Here's a sample `requestAttributes` response.

```
CommunityUrl=http://my-dev-ed.my.salesforce.com:5555/discover
IpAddress=55.255.0.0
UserAgent=Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_4) AppleWebKit/605.1.15
Platform=Mac OSX
Application=Browser
City=San Mateo
Country=United States
Subdivision=California
```

MyDomainLoginDiscoveryHandler Example Implementation

Here's an example of the `Auth.MyDomainLoginDiscoveryHandler` interface. This sample clas the default logic for My Domain login discovery using password authentication. You can the code to ensure it meets your needs. The requestAttributes parameter provides additi information that you can use in the discovery logic. Attributes include MyDomainUrl, IpA UserAgent, and location information (such as Country and City). Use `Auth.DiscoveryCustomErrorException` to throw custom errors to display on the login page.

To implement this interface, the My Domain login page type must be set to Discovery.

```
// This sample class contains the default logic for My Domain login discovery b
// You can customize the code to ensure it meets your needs. The requestAttribu
// provides additional information you can use in the discovery logic. Attribut
// IpAddress, UserAgent, and location information (such as Country and City).
// Use Auth.DiscoveryCustomErrorException to throw custom errors which will be
 global class MyDomainDiscLoginDefaultHandler implements Auth.MyDomainLoginDisc
 global PageReference login(String identifier, String startUrl, Map<String, Str
{
   if (identifier != null) {
        // Search for user by email
        List<User> users = [SELECT Id FROM User WHERE Email = :identifier AND I
        if (!users.isEmpty() && users.size() == 1) {
            return discoveryResult(users[0], startUrl, requestAttributes);
        } else {
            throw new Auth.LoginDiscoveryException('No unique user found. User
        }
```

```
    // To get the URL for a My Domain subdomain, you can pass null in the commu
    // String ssoUrl = Auth.AuthConfiguration.getSamlSsoUrl(null, startUrl, SSO
    // return new PageReference(ssoUrl);
    return null;
  }
     private PageReference discoveryResult(User user, String startUrl, Map<Stri
     {
     PageReference ssoRedirect = getSsoRedirect(user, startUrl, requestAttribut
      if (ssoRedirect != null) {
        return ssoRedirect;
      }
      else {
        return Auth.SessionManagement.finishLoginDiscovery(Auth.LoginDiscoveryM
      }
   }
}
```

Test Class for MyDomainDiscLoginDefaultHandler Class

The following is the test class for MyDomainDiscoveryLoginHandler. For the test to work, must have the My Domain login page type set to Discovery.

```
// Test class for MyDomainDiscLoginDefaultHandler
@isTest
class MyDomainDiscLoginDefaultHandlerTest {
    /* Test Discoverable handler login.
       Create a user with specific email identifier and invoke login.
       Expected : User should be discovered and pagereference should be returne
     */
    @isTest static void testLogin() {
        // Create user
        String identifierEmail = getUniqueName() + '@test.org';
        createTestUser(identifierEmail);
        Map<String, String> requestAttributes = new Map<String, String>();
        String startUrl = '';
        MyDomainDiscLoginDefaultHandler myDomainDiscLoginDefaultHandler = new M
        // Invoke login method from handler with the email of user created
        PageReference  pageReference = myDomainDiscLoginDefaultHandler.login(id
        // Asser page reference is returned
        System.assertNotEquals(null, pageReference, 'Page reference was not ret
    }
    /* Test Discoverable handler login with invalid (non-existing) user.
        Expected : Auth.LoginDiscoveryException
     */
    @isTest static void testLoginWithInvalidUser() {
        try {
            Map<String, String> requestAttributes = new Map<String, String>();
            String startUrl = '';
            String uniqueName = getUniqueName();
            String email = uniqueName + '@test.org';
            MyDomainDiscLoginDefaultHandler myDomainDiscLoginDefaultHandler = n
            // Invoke login method from handler with non-existing user
            myDomainDiscLoginDefaultHandler.login(email, startUrl, requestAttri
        }catch (Auth.LoginDiscoveryException loginDiscoveryException) {
            // Assert exception message
            System.assert(loginDiscoveryException.getMessage().contains('No uni
        }
    }
    /*
       Generate a random name
     */
    private static String getUniqueName() {
        String orgId = UserInfo.getOrganizationId();
        String dateString = String.valueof(Datetime.now()).replace(' ','').repl
        Integer randomInt = Integer.valueOf(math.rint(math.random()*1000000));
        String uniqueName = orgId + dateString + randomInt;
        return uniqueName;
```

```
            String uniqueName = getUniqueName();
            Profile pf = [SELECT Id FROM Profile WHERE Name='Standard User'];
            String profileID = pf.Id;
            String fName = 'fname';
            String lName = uniqueName + '-lname';
            User tuser = new User(  firstname = fName,
                                    lastName = lName,
                                    email = identifierEmail,
                                    Username = uniqueName + '@test.org',
                                    EmailEncodingKey = 'ISO-8859-1',
                                    Alias = uniqueName.substring(18, 23),
                                    TimeZoneSidKey = 'America/Los_Angeles',
                                    LocaleSidKey = 'en_US',
                                    LanguageLocaleKey = 'en_US',
                                    ProfileId = profileID);
            insert tuser;
        }
    }
```

DID THIS ARTICLE SOLVE YOUR ISSUE?
Let us know so we can improve!

Share your