Developers

Site Class

Apex Reference Guide / System Namespace / OrgLimits Class

Site Class

Use the `Site` Class to manage your sites. Change, reset, validate, and check the expiration of passwords. Create site users, person accounts, and portal users. Get the admin email and ID. Get various URLs, the path prefix, the ID, the template, and the type of the site. Log in to the site.

Namespace

System

Site Methods

The following are methods for `Site`. All methods are static.

- changePassword(newPassword, verifyNewPassword, oldPassword)
  Changes the password of the current user.
- createExternalUser(user, accountId)
  Creates a Salesforce Site or Experience Cloud site user for the given account and associates it with the site.
- createExternalUser(user, accountId, password)
  Creates a Salesforce Site or Experience Cloud site user for the given account and associates it with the site. This method sends an email with the specified password to the user.
- createExternalUser(user, accountId, password, sendEmailConfirmation)
  Creates a Salesforce Site or Experience Cloud site user and associates it with the given account. This method sends the user an email with the specified password and a new user confirmation email.
- createPersonAccountPortalUser(user, ownerId, password)
  Creates a person account using the default record type defined on the guest user's profile, then enables it for the site's portal.
- createPersonAccountPortalUser(user, ownerId, recordTypeId, password)
  Creates a person account using the specified *recordTypeID*, then enables it for the site's portal.
- createPortalUser(user, accountId, password, sendEmailConfirmation)
  Creates a portal user for the given account and associates it with the site's portal.
- forgotPassword(username, emailTemplateName)
  Resets the user's password and sends an email to the user with the user’s new password. You can specify a custom email template or use the default email template. Returns a value indicating whether the password reset was successful.
- forgotPassword(username)
  Resets the user's password and sends an email to the user with the user’s new password. Returns a value indicating whether the password reset was successful.
- getAdminEmail()
  Returns the email address of the site administrator.
- getAdminId()
  Returns the user ID of the site administrator.
- getAnalyticsTrackingCode()
  The tracking code associated with your site. Services such as Google Analytics can use this code to track page request data for your site.

Returns a base URL for the current site that doesn't use a Force.com subdomain. The returned URL uses the same protocol (HTTP or HTTPS) as the current request if at least one non-Force.com custom URL that supports HTTPS exists on the site. The returned value never ends with a / character. If all the custom URLs in this site end in Force.com or this site has no custom URLs, then this returns an empty string. If the current request is not a site request, then this method returns an empty string. This method replaced getCustomWebAddress and includes the custom URL's path prefix..

- getBaseInsecureUrl()
  Deprecated. Returns a base URL for the current site that uses HTTP instead of HTTPS. The current request's domain is used. The returned value includes the path prefix and never ends with a / character. If the current request is not a site request, then this method returns an empty string.
- getBaseRequestUrl()
  Returns the base URL of the current site for the requested URL. This isn't influenced by the referring page's URL. The returned URL uses the same protocol (HTTP or HTTPS) as the current request. The returned value includes the path prefix and never ends with a / character. If the current request is not a site request, then this method returns an empty string.
- getBaseSecureUrl()
  Returns a base URL for the current site that uses HTTPS instead of HTTP. The current request's domain is preferred if it supports HTTPS. Domains that are not Force.com subdomains are preferred over Force.com subdomains. A Force.com subdomain, if associated with the site, is used if no other HTTPS domains exist in the current site. If no HTTPS custom URLs exist in the site, then this method returns an empty string. The returned value includes the path prefix and never ends with a / character. If the current request is not a site request, then this method returns an empty string.
- getBaseUrl()
  Returns the base URL of the current site that references and links should use. Note that this field may return the referring page's URL instead of the current request's URL. The returned value includes the path prefix and never ends with a / character. If the current request is not a site request, then this field returns an empty string. This field replaces getCurrentSiteUrl.
- getCustomWebAddress()
  Deprecated. This method was replaced by `getBaseCustomUrl()` in API version 30.0.
- getDomain()
  Returns your Salesforce Sites based URL.
- getErrorDescription()
  Returns the error description for the current page if it's a designated error page for the site and an error exists; otherwise, returns an empty string.
- getErrorMessage()
  Returns an error message for the current page if it's a designated error page for the site and an error exists; otherwise, returns an empty string.
- getExperienceId()
  Returns the value of the experience ID (expid). This expid value comes from a cookie in the user's web browser.
- getMasterLabel()
  Returns the value of the Master Label field for the current site. If the current request is not a site request, then this field returns `null`.
- getName()
  Returns the API name of the current site.
- getOriginalUrl()
  Returns the original URL for this page if it's a designated error page for the site; otherwise, returns `null`.
- getPasswordPolicyStatement()
  Returns the password requirements for a Salesforce Site or Experience Cloud site created with the Customer Service template.

- **getPrefix()**
  Deprecated. This method was replaced by `getPathPrefix()` in API version 30.0.
- **getSiteId()**
  Returns the ID of the current site. If the current request is not a site request, then this field returns `null`.
- **getTemplate()**
  Returns the template name associated with the current site; returns the default template if no template has been designated.
- **getSiteType()**
  Returns the API value of the site type field for the current site. This can be Visualforce for a Salesforce site, Siteforce for a Site.com site, ChatterNetwork for an Experience Cloud site, or ChatterNetworkPicasso for an Experience Cloud site. If the current request is not a site request, then this method returns `null`.
- **getSiteTypeLabel()**
  Returns the value of the Site Type field's label for the current site. If the current request is not a site request, then this method returns `null`.
- **isLoginEnabled()**
  Returns `true` if the current site is associated with an active login-enabled portal; otherwise returns `false`.
- **isPasswordExpired()**
  For authenticated users, returns `true` if the currently logged-in user's password is expired. For non-authenticated users, returns `false`.
- **isRegistrationEnabled()**
  Returns `true` if the current site is associated with an active self-registration-enabled Customer Portal; otherwise returns `false`.
- **isValidUsername(username)**
  Returns `true` if the given username is valid; otherwise, returns `false`.
- **login(username, password, startUrl)**
  Allows users to log in to the current site with the given username and password, then takes them to the `startUrl`. If `startUrl` is not a relative path, it defaults to the site's designated index page.
- **passwordlessLogin(userId, methods, startUrl)**
  Logs in a user to a Salesforce Site or Experience Cloud site using an identity verification method, such as email or text, instead of a password. Passwordless login is a convenient, mobile-centric way to welcome users into your site. Let your users log in with something other than their password, like their email address or phone number.
- **setExperienceId(expIdValue)**
  Sets the experience ID for the current user. Use this method to populate the value of the experience ID (expid) cookie in the user's web browser.
- **setPortalUserAsAuthProvider(user, contactId)**
  Sets the specified user information within the site's portal via an authentication provider.
- **validatePassword(user, password, confirmPassword)**
  Indicates whether a given password meets the requirements specified by org-wide or profile-based password policies in the current user's org.

changePassword(newPassword, verifyNewPassword, oldPassword)

Changes the password of the current user.

Signature

```
public static System.PageReference changePassword(String newPassword, String verifyNewPassword,
String oldPassword)
```

verifyNewPassword

Type: String

oldPassword

Type: String

Optional only if the current user’s password has expired; otherwise, required.

Return Value

Type: System.PageReference

Usage

Calls to this method in API version 30.0 and later can’t commit the transaction automatically. Calls to this method before API version 30.0 commit the transaction, making it impossible to roll back to a save point before the call.

createExternalUser(user, accountId)

Creates a Salesforce Site or Experience Cloud site user for the given account and associates it with the site.

Signature

```
public static Id createExternalUser(SObject user, String accountId)
```

Parameters

user

Type: SObject

Information required to create a user.
The email address of the user is used to look for matching contacts associated with the specified *accountId*. If a matching contact is found and is already used by an external user, self-registration isn’t successful. If a matching contact is found but isn’t used by an external user, it is used for the new external user. If there is no matching contact, a new contact is created for the new external user.

accountId

Type: String

The ID of the account you want to associate the user with.

Return Value

Type: Id

The ID of the user that this method creates.

Usage

This method throws Site.ExternalUserCreateException when user creation fails.

The `nickname` field is required for the `User` sObject when using the `createExternalUser` method.

> **Note**
>
> This method is only valid when a site is associated with a Customer Portal.

createExternalUser(user, accountId, password)

Creates a Salesforce Site or Experience Cloud site user for the given account and associates it with the site. This method sends an email with the specified password to the user.

Signature

```
public static Id createExternalUser(SObject user, String accountId, String password)
```

Parameters

user

Type: SObject

Information required to create a user.
The email address of the user is used to look for matching contacts associated with the specified *accountId*. If a matching contact is found and is already used by an external user, self-registration isn’t successful. If a matching contact is found but isn’t used by an external user, it is used for the new external user. If there is no matching contact, a new contact is created for the new external user.

accountId

Type: String

The ID of the account you want to associate the user with.

password

Type: String

The password of the Salesforce Site or Experience Cloud site user. If not specified, or if set to `null` or an empty string, this method sends a new password email to the portal user.

Return Value

Type: Id

The ID of the user that this method creates.

Usage

This method throws Site.ExternalUserCreateException when user creation fails.

The `nickname` field is required for the `User` sObject when using the `createExternalUser` method.

> **Note**
>
> This method is only valid when a site is associated with a Customer Portal.

Calls to this method in API version 30.0 and later can’t commit the transaction automatically. Calls to this method before API version 30.0 commit the transaction, making it impossible to roll back to a save point before the call.

createExternalUser(user, accountId, password, sendEmailConfirmation)

Creates a Salesforce Site or Experience Cloud site user and associates it with the given account. This method sends the user an email with the specified password and a new user confirmation email.

Signature

user

Type: SObject

Information required to create a user.
The email address of the user is used to look for matching contacts associated with the specified *accountId*. If a matching contact is found and is already used by an external user, self-registration isn’t successful. If a matching contact is found but isn’t used by an external user, it is used for the new external user. If there is no matching contact, a new contact is created for the new external user.

accountId

Type: String

The ID of the account you want to associate the user with.

password

Type: String

The password of the Salesforce Site or Experience Cloud site user. If not specified, or if set to `null` or an empty string, this method sends a new password email to the portal user.

sendEmailConfirmation

Type: Boolean

Determines whether a new user email is sent to the portal user. Set it to `true` to send a new user email to the portal user. The default is `false`, that is, the new user email isn't sent.

Return Value

Type: Id

The ID of the user that this method creates.

Usage

This method throws Site.ExternalUserCreateException when user creation fails.

The `nickname` field is required for the `User` sObject when using the `createExternalUser` method.

> **Note**
>
> This method is only valid when a site is associated with a Customer Portal.

Calls to this method in API version 30.0 and later can’t commit the transaction automatically. Calls to this method before API version 30.0 commit the transaction, making it impossible to roll back to a save point before the call.

createPersonAccountPortalUser(user, ownerId, password)

Creates a person account using the default record type defined on the guest user's profile, then enables it for the site's portal.

Signature

```
public static ID createPersonAccountPortalUser(sObject user, String ownerId, String password)
```

Parameters

user

Type: sObject

Type: String

Return Value

Type: ID

Usage

Calls to this method in API version 30.0 and later can’t commit the transaction automatically. Calls to this method before API version 30.0 commit the transaction, making it impossible to roll back to a save point before the call.

> **Note**
>
> This method is only valid when a site is associated with a Customer Portal, and when the user license for the default new user profile is a high-volume portal user.

createPersonAccountPortalUser(user, ownerId, recordTypeId, password)

Creates a person account using the specified *recordTypeID*, then enables it for the site's portal.

Signature

```
public static ID createPersonAccountPortalUser(sObject user, String ownerId, String
recordTypeId, String password)
```

Parameters

user

Type: sObject

ownerId

Type: String

recordTypeId

Type: String

password

Type: String

Return Value

Type: ID

Usage

Calls to this method in API version 30.0 and later can’t commit the transaction automatically. Calls to this method before API version 30.0 commit the transaction, making it impossible to roll back to a save point before the call.

> **Note**
>
> This method is only valid when a site is associated with a Customer Portal, and when the user license for the default new user profile is a high-volume portal user.

Signature

```
public static ID createPortalUser(sObject user, String accountId, String password, Boolean
sendEmailConfirmation)
```

Parameters

user

Type: sObject

accountId

Type: String

password

Type: String

(Optional) The password of the portal user. If not specified, or if set to `null` or an empty string, this method sends a new password email to the portal user.

sendEmailConfirmation

Type: Boolean

(Optional) Determines whether a new user email is sent to the portal user. Set it to `true` to send a new user email to the portal user. The default is `false`, that is, the new user email isn't sent.

Return Value

Type: ID

Usage

If you're using API version 34.0 or later, we recommend using the `createExternalUser()` methods because they offer better error handling than this method.

The `nickname` field is required for the user sObject when using the `createPortalUser` method.

> **Note**
>
> This method is only valid when a site is associated with a Customer Portal.

Calls to this method in API version 30.0 and later can't commit the transaction automatically. Calls to this method before API version 30.0 commit the transaction, making it impossible to roll back to a save point before the call.

forgotPassword(username, emailTemplateName)

Resets the user's password and sends an email to the user with the user's new password. You can specify a custom email template or use the default email template. Returns a value indicating whether the password reset was successful.

Signature

```
public static Boolean forgotPassword(String username,String emailTemplateName)
```

Parameters

username

Type: String

emailTemplateName

Return Value

Type: Boolean

> **Note**
>
> The return value is always true unless it’s called outside of a Visualforce page.

Usage

Calls to this method in API version 30.0 and later can’t commit the transaction automatically. Calls to this method before API version 30.0 commit the transaction, making it impossible to roll back to a save point before the call.

Calls to this method are subject to rate-limiting. If your rate of calls exceeds the limit, Salesforce doesn't send the password reset email. If you experience this issue, try waiting for an hour before you send another call.

> **Note**
>
> `Site.forgotPassword` cannot be used with the `@future` method, which enables asynchronous execution.

forgotPassword(username)

Resets the user's password and sends an email to the user with the user’s new password. Returns a value indicating whether the password reset was successful.

Signature

```
public static Boolean forgotPassword(String username)
```

Parameters

username

Type: String

Return Value

Type: Boolean

> **Note**
>
> The return value is always true unless it’s called outside of a Visualforce page.

Usage

Calls to this method in API version 30.0 and later can’t commit the transaction automatically. Calls to this method before API version 30.0 commit the transaction, making it impossible to roll back to a save point before the call.

Calls to this method are subject to rate-limiting. If your rate of calls exceeds the limit, Salesforce doesn't send the password reset email. If you experience this issue, try waiting for an hour before you send another call.

> **Note**

getAdminEmail()

Returns the email address of the site administrator.

Signature

```
public static String getAdminEmail()
```

Return Value

Type: String

getAdminId()

Returns the user ID of the site administrator.

Signature

```
public static ID getAdminId()
```

Return Value

Type: ID

getAnalyticsTrackingCode()

The tracking code associated with your site. Services such as Google Analytics can use this code to track page request data for your site.

Signature

```
public static String getAnalyticsTrackingCode()
```

Return Value

Type: String

getCurrentSiteUrl()

Deprecated. This method was replaced by `getBaseUrl()` in API version 30.0. Returns the base URL of the current site that references and links should use.

Signature

Note that this may return the referring page's URL instead of the current request's URL. The returned value includes the path prefix and always ends with a `/` character. If the current request is not a site request, then this method returns `null`. If the current request is not a site request, then this method returns `null`. This method was replaced by getBaseUrl in API version 30.0.

```
public static String getCurrentSiteUrl()
```

Return Value

Type: String

Usage

Use `getBaseUrl()` instead.

getBaseCustomUrl()

empty string. This method replaced getCustomWebAddress and includes the custom URL's path prefix..

Signature

```
public static String getBaseCustomUrl()
```

Return Value

Type: String

Usage

This method replaces `getCustomWebAddress()` and includes the custom URL's path prefix.

getBaseInsecureUrl()

Deprecated. Returns a base URL for the current site that uses HTTP instead of HTTPS. The current request's domain is used. The returned value includes the path prefix and never ends with a `/` character. If the current request is not a site request, then this method returns an empty string.

Signature

```
public static String getBaseInsecureUrl()
```

Return Value

Type: String

getBaseRequestUrl()

Returns the base URL of the current site for the requested URL. This isn't influenced by the referring page's URL. The returned URL uses the same protocol (HTTP or HTTPS) as the current request. The returned value includes the path prefix and never ends with a `/` character. If the current request is not a site request, then this method returns an empty string.

Signature

```
public static String getBaseRequestUrl()
```

Return Value

Type: String

getBaseSecureUrl()

Returns a base URL for the current site that uses HTTPS instead of HTTP. The current request's domain is preferred if it supports HTTPS. Domains that are not Force.com subdomains are preferred over Force.com subdomains. A Force.com subdomain, if associated with the site, is used if no other HTTPS domains exist in the current site. If no HTTPS custom URLs exist in the site, then this method returns an empty string. The returned value includes the path prefix and never ends with a `/` character. If the current request is not a site request, then this method returns an empty string.

Signature

```
public static String getBaseSecureUrl()
```

Return Value

Type: String

request, then this field returns an empty string. This field replaces getCurrentSiteUrl.

Signature

```
public static String getBaseUrl()
```

Return Value

Type: String

Usage

This method replaces `getCurrentSiteUrl()`.

getCustomWebAddress()

Deprecated. This method was replaced by `getBaseCustomUrl()` in API version 30.0.

Signature

Returns the request's custom URL if it doesn't end in Lightning Platform or returns the site's primary custom URL. If neither exist, then this returns null. Note that the URL's path is always the root, even if the request's custom URL has a path prefix. If the current request is not a site request, then this method returns null. The returned value always ends with a `/` character.

```
public static String getCustomWebAddress()
```

Return Value

Type: String

Usage

Use `getBaseCustomUrl()` instead.

getDomain()

Returns your Salesforce Sites based URL.

Signature

```
public static String getDomain()
```

Return Value

Type: String

getErrorDescription()

Returns the error description for the current page if it’s a designated error page for the site and an error exists; otherwise, returns an empty string.

Signature

```
public static String getErrorDescription()
```

Return Value

Type: String

getErrorMessage()

```
public static String getErrorMessage()
```

Return Value

Type: String

getExperienceId()

Returns the value of the experience ID (expid). This expid value comes from a cookie in the user's web browser.

Signature

```
public static String getExperienceId()
```

Return Value

Type: String

Usage

Use the `getExperienceId` and `setExperienceId` methods to implement dynamic login experiences. You can set the experience ID with `setExperienceId` or by extending the following endpoints with `expid_`*`value`*.

- *`community-url`*`/services/oauth2/authorize/expid_`*`value`*
- *`community-url`*`/idp/endpoint/HttpPost/expid_`*`value`*
- *`community-url`*`/idp/endpoint/HttpRedirect/expid_`*`value`*
- *`community-url_login_page`*`/expid={`*`value`*`}`
- *`community-url`*`/CommunitiesSelfReg?expid={`*`value`*`}`
- *`secur`*`/forgotpassword.jsp?expid={`*`value`*`}`

The cookie is set when the browser loads the URLs with the expid values.

getMasterLabel()

Returns the value of the Master Label field for the current site. If the current request is not a site request, then this field returns `null`.

Signature

```
public static String getMasterLabel()
```

Return Value

Type: String

getName()

Returns the API name of the current site.

Signature

```
public static String getName()
```

Return Value

Type: String

getOriginalUrl()

```
public static String getOriginalUrl()
```

Return Value

Type: String

getPasswordPolicyStatement()

Returns the password requirements for a Salesforce Site or Experience Cloud site created with the Customer Service template.

Signature

```
public static String getPasswordPolicyStatement()
```

Return Value

Type: String

getPathPrefix()

Returns the URL path prefix of the current site or an empty string if none. For example, if the requested site URL is `https://myco.my.salesforce-sites.com/partners`, then `/partners` is the path prefix. If the current request is not a site request, then this method returns an empty string. This method replaced getPrefix in API version 30.0.

Signature

```
public static String getPathPrefix()
```

Return Value

Type: String

getPrefix()

Deprecated. This method was replaced by `getPathPrefix()` in API version 30.0.

Signature

Returns the URL path prefix of the current site. For example, if your site URL is `*MyDomainName*.my.salesforce-sites.com/partners`, `/partners` is the path prefix. Returns `null` if the prefix isn’t defined. If the current request is not a site request, then this method returns a `null`.

```
public static String getPrefix()
```

Return Value

Type: String

getSiteId()

Returns the ID of the current site. If the current request is not a site request, then this field returns `null`.

Signature

```
public static String getSiteId()
```

Return Value

Type: Id

Signature

```
public static System.PageReference getTemplate()
```

Return Value

Type: System.PageReference

getSiteType()

Returns the API value of the site type field for the current site. This can be Visualforce for a Salesforce site, Siteforce for a Site.com site, ChatterNetwork for an Experience Cloud site, or ChatterNetworkPicasso for an Experience Cloud site. If the current request is not a site request, then this method returns `null`.

Signature

```
public static String getSiteType()
```

Return Value

Type: String

getSiteTypeLabel()

Returns the value of the Site Type field's label for the current site. If the current request is not a site request, then this method returns `null`.

Signature

```
public static String getSiteTypeLabel()
```

Return Value

Type: String

isLoginEnabled()

Returns `true` if the current site is associated with an active login-enabled portal; otherwise returns `false`.

Signature

```
public static Boolean isLoginEnabled()
```

Return Value

Type: Boolean

isPasswordExpired()

For authenticated users, returns `true` if the currently logged-in user's password is expired. For non-authenticated users, returns `false`.

Signature

```
public static Boolean isPasswordExpired()
```

Return Value

Type: Boolean

Signature

```
public static Boolean isRegistrationEnabled()
```

Return Value

Type: Boolean

isValidUsername(username)

Returns `true` if the given username is valid; otherwise, returns `false`.

Signature

```
public static Boolean isValidUsername(String username)
```

Parameters

username

Type: String

The username to test for validity.

Return Value

Type: Boolean

login(username, password, startUrl)

Allows users to log in to the current site with the given username and password, then takes them to the `startUrl`. If `startUrl` is not a relative path, it defaults to the site's designated index page.

Signature

```
public static System.PageReference login(String username, String password, String startUrl)
```

Parameters

username

Type: String

password

Type: String

startUrl

Type: String

Return Value

Type: System.PageReference

Usage

All DML statements before the call to `Site.login` get committed. It’s not possible to roll back to a save point that was created before a call to `Site.login`.

> **Note**
>
> Do not include `http://` or `https://` in the `startURL`.

their email address or phone number.

Signature

`public static System.PageReference passwordlessLogin(Id userId, List<Auth.VerificationMethod> methods, String startUrl)`

Parameters

userId

Type: Id

ID of the user to log in.

methods

Type: List<Auth.VerificationMethod>

List of identity verification methods available to the user for passwordless login.

startUrl

Type: String

Path to the page that users see after they log in.

Return Value

Type: System.PageReference

Usage

Include this method in the Apex controller of a custom login page implementation.

PasswordlessLogin Example

This simple code example of an Apex controller contains the `passwordlessLogin` method. The `PageReference` returned by `passwordlessLogin` redirects the user to the Salesforce Verify page. When the user enters the correct code, the user is redirected to the site page specified by the start URL.

```
global with sharing class MFILoginController
{
   //Input variables
   global String input {get; set;}
   public String startURL  {get; set;}
   public List<Auth.VerificationMethod> methods;
   public String error;

   global MFILoginController()
   {
       // Add verification methods in priority order
       methods = new List<Auth.VerificationMethod>();
       methods.add(Auth.VerificationMethod.SMS);
       methods.add(Auth.VerificationMethod.EMAIL);
       methods.add(Auth.VerificationMethod.U2F);
       methods.add(Auth.VerificationMethod.SALESFORCE_AUTHENTICATOR);
       methods.add(Auth.VerificationMethod.TOTP);
   }

   global PageReference login() {
        List<User> users = null;

        // Empty input
        if(input == null || input == '')
        {
            error = 'Enter Username';
            return null;
```

```
                return null;
            }

            if (startURL == null) startURL = '/';
            return Site.passwordlessLogin(users[0].id, methods, startURL);
        }
    }
```

setExperienceId(expIdValue)

Sets the experience ID for the current user. Use this method to populate the value of the experience ID (expid) cookie in the user’s web browser.

Signature

`public static void setExperienceId(String expIdValue)`

Parameters

expIdValue

Type: String

A value that indicates the user’s login experience.
The value must contain alphanumeric characters only, up to 30 characters.

Usage

Use `setExperienceId` when you’re implementing dynamic login experiences. A login experience refers to a login page plus any secondary pages associated with the login page (such as multi-factor authentication (MFA) or a login flow). You define different login experiences depending on who users are or where they’re logging in from. For example, you can require a different registration process based on the user’s location. In this case, `expIdValue` includes a state or country code. When the user logs in, the URL contains the experience ID parameter, `{expid}`. The `{expid}` parameter is replaced by the value stored in `expIdValue`, such as `.jp`. Then the user is redirected to the Japanese login experience.

Example

```
String expid = ApexPages.currentPage().getParameters().get('expid');
  if (expId != null) {
  Site.setExperienceId(expId);
  }
```

setPortalUserAsAuthProvider(user, contactId)

Sets the specified user information within the site’s portal via an authentication provider.

Signature

`public static Void setPortalUserAsAuthProvider(sObject user, String contactId)`

Parameters

user

Type: sObject

contactId

Type: String

Return Value

- Calls to this method in API version 30.0 and later can’t commit the transaction automatically. Calls to this method before API version 30.0 commit the transaction, making it impossible to roll back to a save point before the call.
- For more information on an authentication provider, see RegistrationHandler.

validatePassword(user, password, confirmPassword)

Indicates whether a given password meets the requirements specified by org-wide or profile-based password policies in the current user’s org.

Signature

```
public static void validatePassword(SObject user, String password, String confirmPassword)
```

Parameters

user

Type: SObject

The user attempting to create a password during self-registration for a Salesforce Site or Experience Cloud site.

password

Type: String

The password entered by the user.

confirmPassword

Type: String

The password reentered by the user to confirm the password.

Return Value

Type: void

Usage

If validation fails when the method is run in a Lightning controller, this method throws an Apex exception describing the failed validation. If validation fails when the method is run in a Visualforce controller, the method provides Visualforce error messages.

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

Share your feedback

Quip AppExchange

Privacy Information Terms of Service Legal Use of Cookies Trust Cookie Preferences

✓✕ Your Privacy Choices Responsible Disclosure Contact