ccrz.cc_hk_SSO

The ccrz.cc_hk_SSO extension allows for certain CloudCraze OOTB links to be overwritten, for example, to link to a Single Sign On service for login.

global virtual Map<String,Object> getLinkOverrideMap(Map<String,Object> inpData)

The getLinkOverrideMap method returns the list of overridden URLs as a map. The returned URLs should be constructed in such a way to include the passed in cart id and CSR query string. The cart id query string parameter should be "cartId".

Inputs

ccrz.cc_hk_SSO.ENCRYPTEDCARTID

The current encrypted cart id (ccrz__E_Cart__c.ccrz__EncryptedId__c). This may be null or blank.

ccrz.cc_hk_SSO.CSRQUERYSTRING

The current CSR query string parameters. This will contain data such as the current portal user id, etc.

Outputs

ccrz.cc_hk_SSO.HEADER_LOGIN

The URL that the login link in the header should point to. Will only apply to anonymous users.

ccrz.cc_hk_SSO.HEADER_LOGOUT

The URL that the logout link in the header should point to. Will only apply to logged in users.

ccrz.cc_hk_SSO.HEADER_MYACCOUNT

The URL that the My Account link the header should point to.

ccrz.cc_hk_SSO.MYACCOUNT_EDITPROFILE

The URL that the user will be taken to when clicking the edit profile button within the My Account section.

ccrz.cc_hk_SSO.MYACCOUNT_CHANGEPASSWORD

The URL that the user will be taken to when clicking the change password link within My Account.

cc_sample_sso

```
global with sharing class cc_sample_sso extends ccrz.cc_hk_SSO{
        public static final String USER_TYPE_GUEST = 'Guest';

         global override Map<String,Object> getLinkOverrideMap
(Map<String,Object> inpData) {
        Map<String,Object> linkOverrideMap = new Map<String,Object>();
        linkOverrideMap.put(HEADER_LOGIN,buildLoginLink(inpData));
        linkOverrideMap.put(HEADER_LOGOUT, false);
        linkOverrideMap.put(HEADER_MYACCOUNT, false);
        linkOverrideMap.put(MYACCOUNT_EDITPROFILE,
buildEditProfileLink());
        linkOverrideMap.put(MYACCOUNT_CHANGEPASSWORD,
```

```
buildChangePasswordLink());
        linkOverrideMap.put(MINICART_LOGIN, false);
        return linkOverrideMap;
    }

/*
* The edit Profile button in CC , should redirect user to the IDP's
Page. This method retrieves the Edit Profile Link from custom settings
and returns that.
*/
 private String buildEditProfileLink(){
            CC_Sample_Integration_Settings__c editProfileCS =
CC_Sample_Integration_Settings__c.getInstance
('SAMPLE_EDIT_PROFILE_LINK');
            if(editProfileCS != null)
                    return editProfileCS.URL__c;
            return '';
    }
/*
* The change Password Tab in CC , should redirect user to the IDP's
Page. This method retrieves the Change Pasword Link from custom
settings and returns that.
*/
    private String buildChangePasswordLink(){
            CC_Sample_Integration_Settings__c changePassCS =
CC_Sample_Integration_Settings__c.getInstance('SAMPLE_CHANGE_PWD_LINK');
            if(changePassCS != null)
                    return changePassCS.URL__c;
            return '';

    }

/*
* This Method builds the login link provided by the Identity Provider
and appends the returnURL to it, so that after login user is redirected
to the same page he was ON.
*/
 private String buildLoginLink(Map<String,Object> inpData){
                if(inpData != null){
                        encryptedCartId = (String)inpData.get(ccrz.
cc_hk_SSO.ENCRYPTEDCARTID);
                        CSRQueryString = (String)inpData.get(ccrz.
cc_hk_SSO.CSRQUERYSTRING);
                }

                String checkoutFlag = 'checkoutflag=false';

                String pageName = getPagename();
                if(pageName != null && pageName.toLowerCase().contains
('cart')){
```

```
                        if(USER_TYPE_GUEST.equals(UserInfo.
getUserType())){
                                checkoutFlag = 'checkoutflag=false';
                        }
                        else{
                                checkoutFlag = 'checkoutflag=true';
                        }
                }
                String startURL = EncodingUtil.urlEncode
('&ccStartURL='+pageName, 'UTF-8');
                String siteprefix = Site.getPrefix();
                if(siteprefix != null)
                {
                        siteprefix = siteprefix.replace('/', '');
                }

                CC_Sample_Integration_Settings__c ssoCustomSetting =
CC_Sample_Integration_Settings__c.getInstance
('SAMPLE_SSO_SETTINGS');
                String SSOURL = ssoCustomSetting.URL__c;
                CC_Sample_Integration_Settings__c
ssoReturnURLCustomSetting = CC_Sample_Integration_Settings__c.
getInstance('SAMPLE_SSO_RETURN_URL');
                String SSOReturnURL = ssoReturnURLCustomSetting.URL__c;

                String returnURL = EncodingUtil.urlencode
(SSOReturnURL,'UTF-8');
                String loginLink = SSOURL + '&TARGET='+ returnURL + '?
cartID='+ encryptedCartId + '%26' +  EncodingUtil.urlEncode
(CSRQueryString, 'UTF-8') + startURL + '&' + checkoutFlag + '&ccstore='
+sitePrefix + ((encryptedCartId != null)? '&ccCartID=' +
encryptedCartId : '');

                return loginLink;

        }

/*
* Gets the Page Name user is ON.
*/
 private static String getPagename() {
                String methodName = 'getPagename ';

                String result = '';
                String pgURL = ApexPages.currentPage().getUrl();

                if(Test.isRunningTest()) {
                        pgURL='/apex/ccrz__'+'HomePage?test=';
                }
                if(pgURL!=null && pgURL.length()>0) {
```

```
                        if(pgURL.contains('?')) {
                                Integer idx=pgURL.indexOf('?');
                                if(idx>0) {
                                        pgURL=pgURL.substring(0, idx);
                                }
                        }
                        result=pgURL;
                }
                return result;
        }


}
```