# Cookbook - SSO

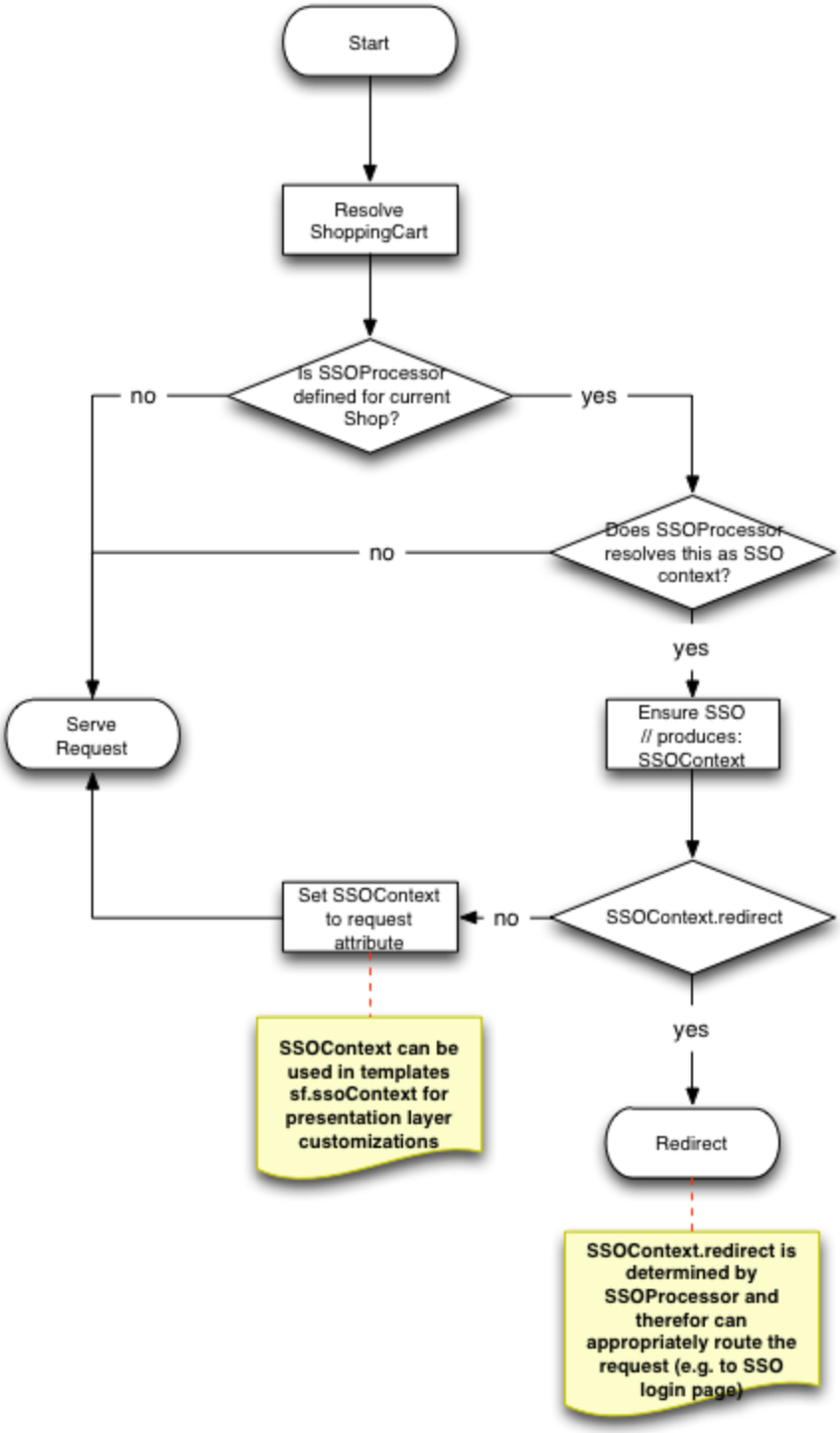

- Overview SaaS 4.1.0+


## Overview SaaS 4.1.0+

The platform offers extension hook in order to facilitate SSO in the frontend, specifically SFG webapp. SSO is handled through ShoppingCartFilterSSOBridge, which allows ShoppingCartFilter to extend the cart resolution mechanism through SSOProcessor implementations.

ShoppingCartFilterSSOBridge in itself is a ConfigurationRegistry and therefore can be configured through the extension mechanism without the need to amend the code (see SSOProcessor).

SSOProcessor interface supports the following flow




SSOProcessor determines whether current request is an SSO and pre-processes the request creating necessary SSOContext object which either can instruct application to redirect to a new location (e.g. SSO login or user migration page) or enrich the request attribute (yc-sso-context) with the SSO Context object that is available in custom controllers and presentation layer though sf.ssoContext variable.

Responsibility of the SSOProcessor is perform SSO operations on the cart object to manage the session. SSOProcessor can trigger core (as well as customised) login/logout/expire cart command to control this.

> SSOProcessor intercepts all URIs that cart filter does and therefore it is recommended to exclude processing of the following URIs: /resources, /imgvault, /filevault and /rest as thouse are used after the main request is served and therefore a valid session is established. Ideally SSOProcessor should only manage main requests (i.e. page rendering requests) and then let the platform session management mechanism handle subsequent requests using cart logged in state.

SSOContext.redirect property allows to any location including absolute URLs to external systems. It is up to SSOProcessor implementation to determine the correct redirect flow for the SSO customer journey.

Custom code in SSOProcessor may intercept /login calls and forward user to the SSO login page hosted outside of the platform.