nslCookie or String consisting of a cookie line in a The ProfileUI class handles displaying all the Profile information to the User storage file. It can then output the cookie in numerous Interface. This class enables you to change the way the profiles are displayed or presented to the user without affecting the backend. This class --getCookieString-The string style needed by will call the registered function when the user selects a new profile CookieService.setCookieString() --getCookieUrl- URI used by CookieService.setCookieString() -getCookieFileString-The string style used for storing cookie in a file <<singleton>> ProfileUI <<static>> instance : ProfileUI* profileSelectedCallback profileSeparator Cookie cookieStatusBar cookieElemList Cookie(cookie : nslCookie) ◆ProfileUI() ◆Cookie(fileString: String) <<static>> getInstance() : ProfileUI* ◆getCookieString(): String <<static>> profileSelected(menuitem) ◆getCookieUrl(): nsIURI ◆addProfileToList(profileName : char*, profileId : int, activeStatus ... ◆getCookieFileString(): Sti... ◆removeProfileFromList(profileId : int) showProfileAsActive(profileId : int) ◆registerProfileSelectedCallback(func_ptr) getMenuItem() uncheckAll() CookieProfile 🃤fileName : nsIFile <<singleton>> CookieProfileContainer CookieProfile(fileName : nslFile) <<<file>>>> profileArray : Array copyToBrowser() activeProfileId : int cookieSwap.js ◆copyFromBrower() profileDir : nsIFile <-static>> instance : CookieProfileContai. ◆getFileHandle() : nsIFile ◆turnOnDebug() ◆removeAllCookies() ◆getNumOfProfiles(): int profileChangeSelecte.. •getProfileName(profileId : int) : String ♦init() ◆getProfile(profileId : int) : CookieProfile This class abstracts the idea of how a profile ◆addProfile(profileName : String) : int ◆removeProfile(profileId : int) is persistently stored. This class also knows how to copy cookies to/from the browser. <<static>> getInstance() getActiveProfileId(): int This represents the collection of setActiveProfileId(newActiveProfileId : int)... functions that glue all the classes together. It is the high level code that coordinates the cookie swaps. The CookieProfileContainer class handles keeping track of where all the CookieProfiles are stored and determining which profile is active. It also enables the user to add/remove profiles.

The Cookie class provides the ability to input a