Class SigUtils Python

This class is a utility class with static methods for calculating and validating cryptographic signatures.

Method Summary

Method		Description
static Boolean	**validateUserSignature**(string UID, string timestamp, string secret, string signature)	Use this method to verify the authenticity of a socialize.getUserInfo API method response, to make sure it is in fact originating from Gigya, and prevent fraud. The socialize.getUserInfo API method response data include the following fields: UID, signatureTimestamp (a timestamp) and UIDSignature (a cryptographic signature). Pass these fields as the corresponding parameters of the validateUserSignature method, along with your partner's "**Secret Key**". Your secret key (provided in BASE64 encoding) is located at the bottom of the Dashboard section on Gigya's website. The return value of the method indicates if the signature is valid (thus, originating from Gigya) or not. If you do not have access to the Partner secret, you can use exchangeUIDSignature to generate a new UIDSignature that is able to be verified with a userKey secret or application secret instead. The return value of the method indicates if the signature is valid (thus, originating from Gigya) or not. Properties (standard): • **UID** : User's UID • **timestamp** : signatureTimestamp • **secret** : Partner secret • **signature** : UIDSignature Properties (when using accounts.exchangeUIDSignature): • **UID** : User's UID • **timestamp** : signatureTimestamp returned from exchangeUIDSignature • **secret** : The userKey secret or application secret used with exchangeUIDSignature • **signature** : The UIDSignature returned from exchangeUIDSignature **validateUserSignature** is only necessary when processing client-to-server calls (where the data in question was received from Gigya to a client and then passed from that client to your server). Server-to-server calls made directly between your server and Gigya do not receive the UIDSignature or signatureTimestamp properties.
static Boolean	**validateFriendSignature**(string UID, string timestamp, string friendUID, string secret, string signature)	Use this method to verify the authenticity of a socialize.getFriendsInfo API method response, to make sure it is in fact originating from Gigya, and prevent fraud. The socialize.getFriendsInfo API method response data include the following fields: UID, signatureTimestamp (a timestamp) and friendshipSignature (a cryptographic signature). Pass these fields as the corresponding parameters of the validateUserSignature method, along with your partner's "**Secret Key**". Your secret key (provided in BASE64 encoding) is located at the bottom of the Dashboard section on Gigya's website. The return value of the method indicates if the signature is valid (thus, originating from Gigya) or not.

static string	**calcSignature**(string baseString, string key)	This is a utility method for generating a HMAC-SHA1 signature.
static string	**getDynamicSessionSignature**(string gltCookie, int timeoutInSeconds, string secret)	This is a utility method for generating the cookie value of a dynamic session expiration cookie. Use this method as part of implementing dynamic control over login session expiration, in conjunction with assigning the value '-1' to the ***sessionExpiration*** parameter of the client side login methods (i.e. showLoginUI / login). Learn more in the Control Session Expiration guide. This method's parameters: • gltCookie - the login token received from Gigya after successful Login. Gigya stores the token in a cookie named: ***"glt_" +*** • timeoutInSeconds - how many seconds until session expiration. For example, if you would like the session to expire in 5 minutes set this parameter to 300. • secret - your Gigya "**Secret Key**", is provided, in BASE64 encoding, at the bottom of the Dashboard page on the Gigya's website.
static string	**getDynamicSessionSignatureUserSigned**(string gltCookie, int timeoutInSeconds, string userKey, string secret)	This utility is the same as above, **getDynamicSessionSignature**, however, allows the session cookie to be generated with an application key or user key (**$userKey**) and the corresponding application key or user key secret, instead of requiring the partner's secret. This is useful when using GConnectors or for 3rd party applications.