kz.gov.pki.knca.applet

Class MainApplet

java.lang.Object
  java.awt.Component
    java.awt.Container
      java.awt.Panel
        java.applet.Applet
          javax.swing.JApplet
            kz.gov.pki.knca.applet.MainApplet

Главный класс апплета

Constructor Summary

Constructors

Constructor and Description
MainApplet()

Method Summary

Methods

Modifier and Type	Method and Description
ResultWrapper	browseKeyStore(java.lang.String storageName, java.lang.String fileExtension, java.lang.String currentDirectory) Метод открывает диалоговое окно для выбора хранилища ключа
ResultWrapper	createCMSSignature(java.lang.String storageName, java.lang.String storagePath, java.lang.String alias, java.lang.String password, java.lang.String dataToSign, boolean attached) Метод формирует и возвращает электронный цифровой подпись в формате CMS
ResultWrapper	createCMSSignatureFromFile(java.lang.String storageName, java.lang.String storagePath, java.lang.String alias, java.lang.String password, java.lang.String filePath, boolean attached) Метод формирует и возвращает электронную цифровую подпись в формате CMS
ResultWrapper	getHash(java.lang.String data, java.lang.String digestAlgName) Метод возвращает хэш данных по указанному алгоритму хеширование
ResultWrapper	getIssuerDN(java.lang.String storageName, java.lang.String storagePath, java.lang.String alias, java.lang.String password) Метод возвращает данные издателя
ResultWrapper	getKeys(java.lang.String storageName, java.lang.String storagePath, java.lang.String password, java.lang.String type) Метод возвращает список данных ключей из выбранного хранилища
ResultWrapper	getNotAfter(java.lang.String storageName, java.lang.String storagePath, java.lang.String alias, java.lang.String password) Метод возвращает дату окончания действия сертификата
ResultWrapper	getNotBefore(java.lang.String storageName, java.lang.String storagePath, java.lang.String alias, java.lang.String password) Метод возвращает дату начала действия сертификата
java.security.Provider	getProvider() Метод возвращает инициализированный провайдер
ResultWrapper	getRdnByOid(java.lang.String storageName, java.lang.String storagePath, java.lang.String alias, java.lang.String password, java.lang.String oid, int oidIndex) Метод возвращает значение RDN по OID
ResultWrapper	getSubjectDN(java.lang.String storageName, java.lang.String storagePath, java.lang.String alias, java.lang.String password) Метод возвращает данные субъекта
void	init() Метод вызывает функции из JavaScript
void	javaScriptCall(java.lang.String functionName, java.lang.Object[] arguments)
ResultWrapper	loadSlotList(java.lang.String storageName) Метод возвращает список имен активных токенов
void	setProvider(java.security.Provider provider)
ResultWrapper	showFileChooser(java.lang.String fileExtension, java.lang.String currentDirectory) Метод открывает диалоговое окно для выбора файла
ResultWrapper	signPlainData(java.lang.String storageName, java.lang.String storagePath, java.lang.String alias, java.lang.String password, java.lang.String dataToSign) Метод подписывает простые данные и возвращает электронную цифровую подпись
ResultWrapper	signXml(java.lang.String storageName, java.lang.String storagePath,

	java.lang.String alias, java.lang.String password, java.lang.String xmlToSign) Метод формирует и возвращает электронный цифровой подпись в формате XML
ResultWrapper	signXmlByElementId(java.lang.String storageName, java.lang.String storagePath, java.lang.String alias, java.lang.String password, java.lang.String xmlToSign, java.lang.String elementId, java.lang.String signatureParentElement) Метод формирует и возвращает электронный цифровой подпись по идентификатору элемента в формате XML
ResultWrapper	verifyCMSSignature(java.lang.String sigantureToVerify, java.lang.String signedData) Метод проверяет валидность электронной цифровой подписи в формате CMS
ResultWrapper	verifyCMSSignatureFromFile(java.lang.String signatureToVerify, java.lang.String filePath) Метод проверяет валидность электронной цифровой подписи в формате CMS
ResultWrapper	verifyPlainData(java.lang.String storageName, java.lang.String storagePath, java.lang.String alias, java.lang.String password, java.lang.String dataToVerify, java.lang.String base64EcodedSignature) Метод проверяет валидность подписанных данных
ResultWrapper	verifyXml(java.lang.String xmlSignature) Метод проверяет валидность электронной цифровой подписи в формате XML
ResultWrapper	verifyXml(java.lang.String xmlSignature, java.lang.String signatureElement) Метод проверяет валидность электронной цифровой подписи в формате XML по идентификатору элемента

Constructor Detail

MainApplet

```
public MainApplet()
```

Method Detail

init

```
public void init()
```

Overrides:

`init` in class `java.applet.Applet`

getProvider

```
public java.security.Provider getProvider()
```

Метод возвращает инициализированный провайдер

Returns:

Провайдер

See Also:

`Provider`

setProvider

```
public void setProvider(java.security.Provider provider)
```

javaScriptCall

```
public void javaScriptCall(java.lang.String functionName,
                  java.lang.Object[] arguments)
```

Метод вызывает функции из `JavaScript`

Parameters:

`functionName` - наименование функции

`arguments` - аргументы функции

loadSlotList

```
public ResultWrapper loadSlotList(java.lang.String storageName)
```

Метод возвращает список имен активных токенов

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken - `Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

Returns:

Имена токенов, разделенных `AppletConstants.KEY_LIST_SEPERATOR` обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`.

showFileChooser

```
public ResultWrapper showFileChooser(java.lang.String fileExtension,
                              java.lang.String currentDirectory)
```

Метод открывает диалоговое окно для выбора файла

Parameters:

`fileExtension` - Расширение для выбора файла

`currentDirectory` - Путь выбираемого файла

Returns:

Полный путь выбранного файла обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`.

browseKeyStore

```
public ResultWrapper browseKeyStore(java.lang.String storageName,
                            java.lang.String fileExtension,
                            java.lang.String currentDirectory)
```

Метод открывает диалоговое окно для выбора хранилища ключа

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken - `Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

`fileExtension` - Расширение для выбора файла

`currentDirectory` - Путь указанный при подаче заявки

Returns:

Полный путь выбранного файла или выбранный носитель ключа обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`.

getKeys

```
public ResultWrapper getKeys(java.lang.String storageName,
                     java.lang.String storagePath,
                     java.lang.String password,
                     java.lang.String type)
```

Метод возвращает список данных ключей из выбранного хранилища

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken - `Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

`storagePath` - Полный путь выбранного файла (*.p12) или выбранное наименование токена/смарт карты.

`password` - Пароль от хранилища ключей.

`type` - Тип ключа: `AppletConstants.KEY_TYPE_AUTH` для получения списка данных ключей для аутентификации; `AppletConstants.KEY_TYPE_SIGN` для получения списка данных ключей для подписи; или любую другую строку для получения списка данных всех ключей.

Returns:

Список данных ключей обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`. Если `ResultWrapper.getErrorCode() == "WRONG_PASSWORD"` возвращает количество оставшихся попыток обернутый в объект `ResultWrapper.getResult()`. Если количество оставшихся попыток меньше нуля, то количество не определено или их бесконечное множество.

getNotAfter

```
public ResultWrapper getNotAfter(java.lang.String storageName,
                         java.lang.String storagePath,
                         java.lang.String alias,
```

java.lang.String password)

Метод возвращает дату окончания действия сертификата

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken - `Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

`storagePath` - Полный путь выбранного файла (*.p12) или выбранное наименование токена/смарт карты.

`alias` - Alias ключа

`password` - Пароль от хранилища ключей

Returns:

Дата окончания действия сертификата обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`. Если `ResultWrapper.getErrorCode() == "WRONG_PASSWORD"` возвращает количество оставшихся попыток обернуты в объект `ResultWrapper.getResult()`. Если количество оставшихся попыток меньше нуля, то количество не определено или их бесконечное множество.

getNotBefore

```
public ResultWrapper getNotBefore(java.lang.String storageName,
                        java.lang.String storagePath,
                        java.lang.String alias,
                        java.lang.String password)
```

Метод возвращает дату начала действия сертификата

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken - `Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

`storagePath` - Полный путь выбранного файла (*.p12) или выбранное наименование токена/смарт карты.

`alias` - Alias ключа

`password` - Пароль от хранилища ключей

Returns:

Дата начала действия сертификата обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`. Если `ResultWrapper.getErrorCode() == "WRONG_PASSWORD"` возвращает количество оставшихся попыток обернуты в объект `ResultWrapper.getResult()`. Если количество оставшихся попыток меньше нуля, то количество не определено или их бесконечное множество.

getSubjectDN

```
public ResultWrapper getSubjectDN(java.lang.String storageName,
                        java.lang.String storagePath,
                        java.lang.String alias,
                        java.lang.String password)
```

Метод возвращает данные субъекта

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken - `Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

`storagePath` - Полный путь выбранного файла (*.p12) или выбранное наименование токена/смарт карты.

`alias` - Alias ключа

`password` - Пароль от хранилища ключей

Returns:

Данные субъекта обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`. Если `ResultWrapper.getErrorCode() == "WRONG_PASSWORD"` возвращает количество оставшихся попыток обернуты в объект `ResultWrapper.getResult()`. Если количество оставшихся попыток меньше нуля, то количество не определено или их бесконечное множество.

getIssuerDN

```
public ResultWrapper getIssuerDN(java.lang.String storageName,
                        java.lang.String storagePath,
                        java.lang.String alias,
                        java.lang.String password)
```

Метод возвращает данные издателя

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken - `Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

`storagePath` - Полный путь выбранного файла (*.p12) или выбранное наименование токена/смарт карты.

`alias` - Alias ключа

`password` - Пароль от хранилища ключей

Returns:

Данные издателя обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`. Если `ResultWrapper.getErrorCode() == "WRONG_PASSWORD"` возвращает количество оставшихся попыток обернуты в объект `ResultWrapper.getResult()`. Если количество оставшихся попыток меньше нуля, то количество не определено или их бесконечное множество.

getRdnByOid

```
public ResultWrapper getRdnByOid(java.lang.String storageName,
                        java.lang.String storagePath,
                        java.lang.String alias,
                        java.lang.String password,
                        java.lang.String oid,
                        int oidIndex)
```

Метод возвращает значение RDN по OID

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken - `Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

`storagePath` - Полный путь выбранного файла (*.p12) или выбранное наименование токена/смарт карты.

`alias` - Alias ключа

`password` - Пароль от хранилища ключей

`oid` - Идентификатор объекта

`oidIndex` - Индекс идентификатора объекта

Returns:

Значение RDN обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`. Если `ResultWrapper.getErrorCode() == "WRONG_PASSWORD"` возвращает количество оставшихся попыток обернуты в объект `ResultWrapper.getResult()`. Если количество оставшихся попыток меньше нуля, то количество не определено или их бесконечное множество.

signPlainData

```
public ResultWrapper signPlainData(java.lang.String storageName,
                          java.lang.String storagePath,
                          java.lang.String alias,
                          java.lang.String password,
                          java.lang.String dataToSign)
```

Метод подписывает простые данные и возвращает электронную цифровую подпись

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken - `Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

`storagePath` - Полный путь выбранного файла (*.p12) или выбранное наименование токена/смарт карты.

`alias` - Alias ключа

`password` - Пароль от хранилища ключей

`dataToSign` - Данные для подписи

Returns:

Драфтовая подпись в формате Base64 обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`. Если `ResultWrapper.getErrorCode() == "WRONG_PASSWORD"` возвращает количество оставшихся попыток обернуты в объект `ResultWrapper.getResult()`. Если количество оставшихся попыток меньше нуля, то количество не определено или их бесконечное множество.

verifyPlainData

```
public ResultWrapper verifyPlainData(java.lang.String storageName,
                            java.lang.String storagePath,
                            java.lang.String alias,
                            java.lang.String password,
                            java.lang.String dataToVerify,
                            java.lang.String base64EcodedSignature)
```

Метод проверяет валидность подписанных данных

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken -

`Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

`storagePath` - Полный путь выбранного файла (*.p12) или выбранное наименование токена/смарт карты.

`alias` - Alias ключа

`password` - Пароль от хранилища ключей

`dataToVerify` - Данные для проверки

`base64EcodedSignature` - Подпись для проверки

Returns:

Результат проверки в формате boolean обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`. Если `ResultWrapper.getErrorCode() == "WRONG_PASSWORD"` возвращает количество оставшихся попыток обернуты в объект `ResultWrapper.getResult()`. Если количество оставшихся попыток меньше нуля, то количество не определено или их бесконечное множество.

createCMSSignature

```
public ResultWrapper createCMSSignature(java.lang.String storageName,
                                java.lang.String storagePath,
                                java.lang.String alias,
                                java.lang.String password,
                                java.lang.String dataToSign,
                                boolean attached)
```

Метод формирует и возвращает электронный цифровой подпись в формате CMS

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken - `Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

`storagePath` - Полный путь выбранного файла (*.p12) или выбранное наименование токена/смарт карты.

`alias` - Alias ключа

`password` - Пароль от хранилища ключей

`dataToSign` - Данные для подписи

`attached` - TODO

Returns:

CMS подпись обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`. Если `ResultWrapper.getErrorCode() == "WRONG_PASSWORD"` возвращает количество оставшихся попыток обернуты в объект `ResultWrapper.getResult()`. Если количество оставшихся попыток меньше нуля, то количество не определено или их бесконечное множество.

createCMSSignatureFromFile

```
public ResultWrapper createCMSSignatureFromFile(java.lang.String storageName,
                                    java.lang.String storagePath,
                                    java.lang.String alias,
                                    java.lang.String password,
                                    java.lang.String filePath,
                                    boolean attached)
```

Метод формирует и возвращает электронную цифровую подпись в формате CMS

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken - `Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

`storagePath` - Полный путь выбранного файла (*.p12) или выбранное наименование токена/смарт карты.

`alias` - Alias ключа

`password` - Пароль от хранилища ключей

`filePath` - Путь к файлу, который необходимо подписать

`attached` - TODO

Returns:

CMS подпись обернутая в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`. Если `ResultWrapper.getErrorCode() == "WRONG_PASSWORD"` возвращает количество оставшихся попыток обернуты в объект `ResultWrapper.getResult()`. Если количество оставшихся попыток меньше нуля, то количество не определено или их бесконечное множество.

verifyCMSSignature

```
public ResultWrapper verifyCMSSignature(java.lang.String sigantureToVerify,
                                java.lang.String signedData)
```

Метод проверяет валидность электронной цифровой подписи в формате CMS

Parameters:

`sigantureToVerify` - Подпись CMS

`signedData` - Исходные данные

Returns:

Результат проверки в формате boolean обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`

verifyCMSSignatureFromFile

```
public ResultWrapper verifyCMSSignatureFromFile(java.lang.String signatureToVerify,
                                                java.lang.String filePath)
```

Метод проверяет валидность электронной цифровой подписи в формате CMS

Parameters:

`signatureToVerify` - Подпись CMS

`filePath` - Путь к файлу с исходными данными

Returns:

Результат проверки в формате boolean обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`

signXml

```
public ResultWrapper signXml(java.lang.String storageName,
                             java.lang.String storagePath,
                             java.lang.String alias,
                             java.lang.String password,
                             java.lang.String xmlToSign)
```

Метод формирует и возвращает электронный цифровой подпись в формате XML

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken - `Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

`storagePath` - Полный путь выбранного файла (*.p12) или выбранное наименование токена/смарт карты.

`alias` - Alias ключа

`password` - Пароль от хранилища ключей

`xmlToSign` - Данные для подписи в формате XML

Returns:

Подписанный XML обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`. Если `ResultWrapper.getErrorCode() == "WRONG_PASSWORD"` возвращает количество оставшихся попыток обернуты в объект `ResultWrapper.getResult()`. Если количество оставшихся попыток меньше нуля, то количество не определено или их бесконечное множество.

signXmlByElementId

```
public ResultWrapper signXmlByElementId(java.lang.String storageName,
                                        java.lang.String storagePath,
                                        java.lang.String alias,
                                        java.lang.String password,
                                        java.lang.String xmlToSign,
                                        java.lang.String elementId,
                                        java.lang.String signatureParentElement)
```

Метод формирует и возвращает электронный цифровой подпись по идентификатору элемента в формате XML

Parameters:

`storageName` - Имя хранилища ключа: для Казтокена - `Storage.KAZTOKEN`; для удостоверния личности - `Storage.KZIDCARD`; для Etoken - `Storage.ETOKEN_72K`; для Jacarta - `Storage.JACARTA`; для файловой системы - `Storage.PKCS12`;

`storagePath` - Полный путь выбранного файла (*.p12) или выбранное наименование токена/смарт карты.

`alias` - Alias ключа

`password` - Пароль от хранилища ключей

`xmlToSign` - Данные для подписи в формате XML

`elementId` - Идентификатор элемента для подписи

`signatureParentElement` - родительский элемент для элемента ds:Signature

Returns:

Подписанный XML обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`. Если `ResultWrapper.getErrorCode() == "WRONG_PASSWORD"` возвращает количество оставшихся попыток обернуты в объект `ResultWrapper.getResult()`. Если количество оставшихся попыток меньше нуля, то количество не определено или их бесконечное множество.

verifyXml

```
public ResultWrapper verifyXml(java.lang.String xmlSignature)
```

Метод проверяет валидность электронной цифровой подписи в формате XML

Parameters:

`xmlSignature` - Подписанный XML

Returns:

Результат проверки в формате boolean обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`

verifyXml

```
public ResultWrapper verifyXml(java.lang.String xmlSignature,
                               java.lang.String signatureElement)
```

Метод проверяет валидность электронной цифровой подписи в формате XML по идентификатору элемента

Parameters:

`xmlSignature` - Подписанный XML

`signatureElement` - Элемент XML с подписью. Если параметр не указан то берется корневой элемент XML

Returns:

Результат проверки в формате boolean обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`

getHash

```
public ResultWrapper getHash(java.lang.String data,
                             java.lang.String digestAlgName)
```

Метод возвращает хэш данных по указанному алгоритму хеширование

Parameters:

`data` - - Данные для хэширование

`digestAlgName` - - Алгоритм хэширование

Returns:

Хэш данных обернутый в объект `ResultWrapper.getResult()`, если `ResultWrapper.getErrorCode() == "NONE"`, или код ошибки `ResultWrapper.getErrorCode()`

kz.gov.pki.knca.applet

Class ResultWrapper

java.lang.Object
    kz.gov.pki.knca.applet.ResultWrapper

Обертка результата или ошибки. Объект данного класса возвращается при вызове методов апплета.

Constructor Summary

Constructors

Constructor and Description
ResultWrapper() Конструктор класса ResultWrapper
ResultWrapper(java.lang.String errorCode) Конструктор класса ResultWrapper
ResultWrapper(java.lang.String errorCode, java.lang.Object result) Конструктор класса ResultWrapper

Method Summary

Methods

Modifier and Type	Method and Description
java.lang.String	**getErrorCode**() Возвращает код ошибки
java.lang.Object	**getResult**() Возвращает результат выполнения метода апплета.
java.lang.Object	**getSecondResult**() Возвращает второй результат выполнения метода апплета, в случае его наличия.
void	**setResult**(java.lang.Object result) Устанавливает результат выполнения метода апплета.
void	**setSecondResult**(java.lang.Object secondResult) Устанавливает второй результат выполнения метода апплета, в случае его наличия.
java.lang.String	**toString**()

Constructor Detail

ResultWrapper

```
public ResultWrapper(java.lang.String errorCode,
             java.lang.Object result)
```

Конструктор класса ResultWrapper

Parameters:

errorCode - код ошибки

result - результат выполнения метода апплета

ResultWrapper

```
public ResultWrapper(java.lang.String errorCode)
```

Конструктор класса ResultWrapper

Parameters:

errorCode - код ошибки

ResultWrapper

```
public ResultWrapper()
```

Конструктор класса ResultWrapper

Method Detail

getResult

```
public java.lang.Object getResult()
```

Возвращает результат выполнения метода апплета.

Returns:

getErrorCode

```
public java.lang.String getErrorCode()
```

Возвращает код ошибки

Returns:

Возвращает код ошибки, и NONE в случае успешного выполнения метода апплета.

getSecondResult

```
public java.lang.Object getSecondResult()
```

Возвращает второй результат выполнения метода апплета, в случае его наличия. Используется при генерации пары ключей.

Returns:

второй результат выполнения метода апплета

setSecondResult

```
public void setSecondResult(java.lang.Object secondResult)
```

Устанавливает второй результат выполнения метода апплета, в случае его наличия. Используется при генерации пары ключей.

Parameters:

`secondResult` - второй результат выполнения метода апплета

setResult

```
public void setResult(java.lang.Object result)
```

Устанавливает результат выполнения метода апплета.

Parameters:

`result` - результат выполнения метода апплета

toString

```
public java.lang.String toString()
```

Overrides:

`toString` in class `java.lang.Object`

kz.gov.pki.kalkan

Enum Storage

java.lang.Object
    java.lang.Enum<Storage>
        kz.gov.pki.kalkan.Storage

Все ключевые хралилища, которые поддерживаются Java криптопровайдером НУЦ РК.

Enum Constant Summary

Enum Constants

Enum Constant and Description
ETOKEN_72K name = "AKEToken72KStore", token = true
JACARTA name = "AKJaCartaStore", token = true
JKS name = "JKS", token = false
KAZTOKEN name = "AKKaztokenStore", token = true
KZIDCARD name = "AKKZIDCardStore", token = true
PKCS12 name = "PKCS12", token = false

Method Summary

Methods

Modifier and Type	Method and Description
`static Storage`	**get**(java.lang.String storageName) Возвращает хранилище по наименванию хранилища.
`java.lang.String`	**getName**() Возвращает наименование хралища
`boolean`	**isToken**() Возвращает положительный результат, если хранилище является токеном, отрицательный - если файловое хранилище.
`static Storage`	**valueOf**(java.lang.String name) Returns the enum constant of this type with the specified name.
`static Storage[]`	**values**() Returns an array containing the constants of this enum type, in the order they are declared.

Enum Constant Detail

KAZTOKEN

```
public static final Storage KAZTOKEN
```

name = "AKKaztokenStore", token = true

KZIDCARD

```
public static final Storage KZIDCARD
```

name = "AKKZIDCardStore", token = true

ETOKEN_72K

```
public static final Storage ETOKEN_72K
```

name = "AKEToken72KStore", token = true

JACARTA

`public static final Storage JACARTA`

name = "AKJaCartaStore", token = true

JKS

`public static final Storage JKS`

name = "JKS", token = false

PKCS12

`public static final Storage PKCS12`

name = "PKCS12", token = false

Method Detail

values

`public static Storage[] values()`

Returns an array containing the constants of this enum type, in the order they are declared. This method may be used to iterate over the constants as follows:

```
for (Storage c : Storage.values())
    System.out.println(c);
```

Returns:

an array containing the constants of this enum type, in the order they are declared

valueOf

`public static Storage valueOf(java.lang.String name)`

Returns the enum constant of this type with the specified name. The string must match *exactly* an identifier used to declare an enum constant in this type. (Extraneous whitespace characters are not permitted.)

Parameters:

`name` - the name of the enum constant to be returned.

Returns:

the enum constant with the specified name

Throws:

`java.lang.IllegalArgumentException` - if this enum type has no constant with the specified name

`java.lang.NullPointerException` - if the argument is null

getName

`public java.lang.String getName()`

Возвращает наименование хралища

isToken

`public boolean isToken()`

Возвращает положительный результат, если хранилище является токеном, отрицательный - если файловое хранилище.

get

`public static Storage get(java.lang.String storageName)`

Возвращает хранилище по наименванию хранилища.

Parameters:

`storageName` - название хранилища

Returns:

Хранилище

kz.gov.pki.knca.applet

Class AppletConstants

java.lang.Object
    kz.gov.pki.knca.applet.AppletConstants

Константные переменные апплета.

Field Summary

Fields

Modifier and Type	Field and Description
static java.lang.String	**BEGIN_NEW_CR** Начало запроса на новый сертификат, value = "-----BEGIN NEW CERTIFICATE REQUEST-----"
static java.lang.String	**END_NEW_CR** Конец запроса на новый сертификат, value = "-----END NEW CERTIFICATE REQUEST-----"
static java.lang.String	**EXT_KEY_USAGE_SSL_CLIENT_OID** OID, который указывает, что сертификат может быть использован в качестве сертификата клиента SSL, value = "1.3.6.1.5.5.7.3.2"
static java.lang.String	**GOSTKZ_FILE_PREFIX** Префикс названия файла с ключом GOST, value = "GOSTKZ_"
static java.lang.String	**JKS_EXTENSION** Расширение ключа: jks, , value = ".jks"
static java.lang.String	**KEY_DETAILS_SEPERATOR** разделитель в списке информации о деталях ключа, value = "<@>"
static java.lang.String	**KEY_LIST_SEPERATOR** разделитель в списке ключей, value = "<:>"
static java.lang.String	**KEY_TYPE_ALL** Тип ключа: все, value = "ALL"
static java.lang.String	**KEY_TYPE_AUTH** Тип ключа: аутентификация, value = "AUTH"
static java.lang.String	**KEY_TYPE_SIGN** Тип ключа: подпись, value = "SIGN"
static java.lang.String	**KEY_TYPE_UNKNOWN** Тип ключа: неизвестный, value = UNKNOWN"
static java.lang.String	**P12_EXTENSION** Расширение ключа: p12, , value = ".p12"
static java.lang.String	**RDN_SEPERATOR** разделитель RDN-ов, value = "<:>"
static java.lang.String	**RSA_FILE_PREFIX** Префикс названия файла с ключом RSA, value = "RSA_"
static java.lang.String	**TOKEN_SEPERATOR** разделитель названия токенов, value = "<:>"
static java.lang.String	**UNKNOWN_SMARTCARD** Неизвестный токен, value = "UNKNOWN_SMARTCARD"
static java.lang.String	**UTF_8_ENCODING** Название кодировки: UTF-8, value = "UTF-8"

Constructor Summary

Constructors

Constructor and Description
AppletConstants()

Field Detail

TOKEN_SEPERATOR

```
public static final java.lang.String TOKEN_SEPERATOR
```

разделитель названия токенов, value = "<:>"

See Also:

Constant Field Values

RDN_SEPERATOR

```
public static final java.lang.String RDN_SEPERATOR
```

разделитель RDN-ов, value = "<:>"

See Also:

Constant Field Values

KEY_LIST_SEPERATOR

```
public static final java.lang.String KEY_LIST_SEPERATOR
```

разделитель в списке ключей, value = "<:>"

See Also:

Constant Field Values

KEY_DETAILS_SEPERATOR

```
public static final java.lang.String KEY_DETAILS_SEPERATOR
```

разделитель в списке информации о деталях ключа, value = "<@>"

See Also:

Constant Field Values

KEY_TYPE_ALL

```
public static final java.lang.String KEY_TYPE_ALL
```

Тип ключа: все, value = "ALL"

See Also:

Constant Field Values

KEY_TYPE_AUTH

```
public static final java.lang.String KEY_TYPE_AUTH
```

Тип ключа: аутентификация, value = "AUTH"

See Also:

Constant Field Values

KEY_TYPE_SIGN

```
public static final java.lang.String KEY_TYPE_SIGN
```

Тип ключа: подпись, value = "SIGN"

See Also:

Constant Field Values

KEY_TYPE_UNKNOWN

```
public static final java.lang.String KEY_TYPE_UNKNOWN
```

Тип ключа: неизвестный, value = UNKNOWN"

See Also:

Constant Field Values

UNKNOWN_SMARTCARD

```
public static final java.lang.String UNKNOWN_SMARTCARD
```

Неизвестный токен, value = "UNKNOWN_SMARTCARD"

See Also:

Constant Field Values

UTF_8_ENCODING

```
public static final java.lang.String UTF_8_ENCODING
```

Название кодировки: UTF-8, value = "UTF-8"

See Also:

Constant Field Values

P12_EXTENSION

```
public static final java.lang.String P12_EXTENSION
```

Расширение ключа: p12, , value = ".p12"

See Also:

Constant Field Values

JKS_EXTENSION

```
public static final java.lang.String JKS_EXTENSION
```

Расширение ключа: jks, , value = ".jks"

See Also:

Constant Field Values

RSA_FILE_PREFIX

```
public static final java.lang.String RSA_FILE_PREFIX
```

Префикс названия файла с ключом RSA, value = "RSA_"

See Also:

Constant Field Values

GOSTKZ_FILE_PREFIX

```
public static final java.lang.String GOSTKZ_FILE_PREFIX
```

Префикс названия файла с ключом GOST, value = "GOSTKZ_"

See Also:

Constant Field Values

BEGIN_NEW_CR

```
public static final java.lang.String BEGIN_NEW_CR
```

Начало запроса на новый сертификат, value = "-----BEGIN NEW CERTIFICATE REQUEST-----"

See Also:

Constant Field Values

END_NEW_CR

```
public static final java.lang.String END_NEW_CR
```

Конец запроса на новый сертификат, value = "-----END NEW CERTIFICATE REQUEST-----"

See Also:

Constant Field Values

EXT_KEY_USAGE_SSL_CLIENT_OID

```
public static final java.lang.String EXT_KEY_USAGE_SSL_CLIENT_OID
```

OID, который указывает, что сертификат может быть использован в качестве сертификата клиента SSL, value = "1.3.6.1.5.5.7.3.2"

See Also:

Constant Field Values

Constructor Detail

AppletConstants

```
public AppletConstants()
```

kz.gov.pki.knca.applet.exception

Enum AECodes

java.lang.Object
  java.lang.Enum<AECodes>
    kz.gov.pki.knca.applet.exception.AECodes

Коды ошибок, которые могут появиться во время выполнения апплета

Enum Constant Summary

Enum Constants

Enum Constant and Description
BAD_XML_FORMAT Неверный формат файла xml
CHANGEPASS_COMMON Общая ошибка при изменении пароля на хранилище
CHANGEPASS_CONTAINS_EMPTY_ALIAS В хранилище не установлен сертификат при изменении пароля на хранилище
CHANGEPASS_EMPTY_NEWPASS Поле новый пароль пустое при изменении пароля на хранилище
CHANGEPASS_INCORRECT_NEWPASS_PATTERN Неверный формат нового пароля при изменении пароля на хранилище
CHANGEPASS_NULL_KEYSTORE Не удалось загрузить хранилище при изменении пароля на хранилище
CHANGEPASS_WRONG_PASSWORD Неверный пароль при изменении пароля на хранилище
COMMON Общая ошибка в апплете
COMMON_CHOOSE_TOKEN_ERROR Ошибка при выборе носителя
EMPTY_KEY_LIST Пустой список ключей
EMPTY_SLOT Устройства не найдены
EXTENSIONS_AUTHORITY_INFORMATION_ACCESS_COMMON Ошибка при генерации расширения: Доступ к сведениям центра сертификации
EXTENSIONS_AUTHORITY_KEY_IDENTIFIER_COMMON Ошибка при генерации расширения: Идентификатор ключа центра сертификации
EXTENSIONS_BASIC_CONSTRAINTS_COMMON Ошибка при генерации расширения: Базовые ограничения
EXTENSIONS_CERTIFICATE_POLICY_COMMON Ошибка при генерации расширения: Политика сертификата
EXTENSIONS_CRL_DISTRIBUTION_POINT_COMMON Ошибка при генерации расширения: Точки распределения списка отзыва
EXTENSIONS_EXTENDED_KEY_USAGE_COMMON Ошибка при генерации расширения: Расширенное использование ключа
EXTENSIONS_FRESHEST_CRL_COMMON Ошибка при генерации расширения: Новейший CRL
EXTENSIONS_KEY_USAGE_COMMON Ошибка при генерации расширения: Использование ключа
EXTENSIONS_KNCA_ADMIN_COMMON Ошибка при генерации расширения: Расширение определяющее администратора НУЦ РК
EXTENSIONS_KNCA_MANAGER_COMMON Ошибка при генерации расширения: Расширение определяющее менеджера НУЦ РК
EXTENSIONS_KNCA_TRUSTED_OPERATOR_COMMON Ошибка при генерации расширения: Расширение определяющее доверенного оператора НУЦ РК
FILE_READ_ERROR Ошибка при чтении файла
GENEXTENSION_CANCEL Отмена действия при генерации расширения
GENKEY_CANCEL Отмена действий при генерации ключа
GENKEY_COMMON Общая ошибка при генерации ключа
GENKEY_EMPTY_DN Пустые поля при генерации ключа

GENKEY_EMPTY_KEYID	Пустой идентификатор ключа при генерации ключа
GENKEY_EMPTY_PKCS10	Пустой запрос (pkcs10) при генерации ключа
GENKEY_INCORRECT_DN_VALUE	Неправильное значение полей при генерации ключа
GENKEY_UNKNOWN_ALG	Неизвестный алгоритм при генерации ключа
GENKEY_WRONG_PASSWORD	Неверный пароль при генерации ключа
GETPUBLICPART_EXCEPTION	Ошибка при загрузке информации с удостоверения личности
LOAD_KEYSTORE_ERROR	Ошибка во время инициализации ключа
LOAD_SLOT_LIST_EXCEPTION	Ошибка при загрузке списка доступных устройств
PRIVILEGED_ACTION_EXCEPTION	Ошибка связанная с доступом
RDN_NOT_FOUND	RDN не найден
SETCERT_CANCEL	Отмена действия при установке сертификата в хранилище
SETCERT_COMMON	Общая ошибка при установке сертификата в хранилище
SETCERT_P12_FILE_NOT_FOUND	Не найден соответствующий файл p12 при установке сертификата в хранилище
SETCERT_WRONG_PASSWORD	Неверный пароль при установке сертификата в хранилище
SIGN_CANCEL	Отмена действия при подписи
SIGN_COMMON	Общая ошибка при подписи
SIGN_EMPTY_PKCS12_STORAGE	В хранилище не установлен сертификат при подписи
SIGN_EMPTY_STORAGE	Пустое хранилище при подписи
SIGN_NULL_KEYSTORE	Не удалось загрузить хранилище при подписи
SIGN_WRONG_PASSWORD	Неверный пароль при подписи
SIGNATURE_ELEMENT_NOT_FOUND_WITHIN_XML	Элемент подписи (ds:Reference) не найден в структуре данного xml
SIGNATURE_VALIDATION_ERROR	Ошибка проверки подписи
TOKEN_CHOOSE_CANCEL	Отмена действия при выборе носителя
UNKNOWN_SIG_ALG	Неизвестный алгоритм подписи
UNKNOWN_SMARTCARD	Неизвестное устройство
UNKNOWN_STORAGE	Неизвестное хранилище
WRONG_PASSWORD	Неверный пароль
XML_PARSE_EXCEPTION	Ошибка при чтении xml файла

Method Summary

Methods

Modifier and Type	Method and Description
`static AECodes`	**valueOf**(`java.lang.String name`) Returns the enum constant of this type with the specified name.
`static AECodes[]`	**values**() Returns an array containing the constants of this enum type, in the order they are declared.

Enum Constant Detail

COMMON

```
public static final AECodes COMMON
```

Общая ошибка в апплете

WRONG_PASSWORD

```
public static final AECodes WRONG_PASSWORD
```

Неверный пароль

LOAD_KEYSTORE_ERROR

```
public static final AECodes LOAD_KEYSTORE_ERROR
```

Ошибка во время инициализации ключа

UNKNOWN_SIG_ALG

```
public static final AECodes UNKNOWN_SIG_ALG
```

Неизвестный алгоритм подписи

GENKEY_UNKNOWN_ALG

```
public static final AECodes GENKEY_UNKNOWN_ALG
```

Неизвестный алгоритм при генерации ключа

GENKEY_EMPTY_DN

```
public static final AECodes GENKEY_EMPTY_DN
```

Пустые поля при генерации ключа

GENKEY_INCORRECT_DN_VALUE

```
public static final AECodes GENKEY_INCORRECT_DN_VALUE
```

Неправильное значение полей при генерации ключа

GENKEY_COMMON

```
public static final AECodes GENKEY_COMMON
```

Общая ошибка при генерации ключа

GENKEY_EMPTY_PKCS10

```
public static final AECodes GENKEY_EMPTY_PKCS10
```

Пустой запрос (pkcs10) при генерации ключа

GENKEY_EMPTY_KEYID

```
public static final AECodes GENKEY_EMPTY_KEYID
```

Пустой идентификатор ключа при генерации ключа

GENKEY_WRONG_PASSWORD

```
public static final AECodes GENKEY_WRONG_PASSWORD
```

Неверный пароль при генерации ключа

GENKEY_CANCEL

```
public static final AECodes GENKEY_CANCEL
```

Отмена действий при генерации ключа

SETCERT_CANCEL

```
public static final AECodes SETCERT_CANCEL
```

Отмена действия при установке сертификата в хранилище

SETCERT_WRONG_PASSWORD

```
public static final AECodes SETCERT_WRONG_PASSWORD
```

Неверный пароль при установке сертификата в хранилище

SETCERT_COMMON

```
public static final AECodes SETCERT_COMMON
```

Общая ошибка при установке сертификата в хранилище

SETCERT_P12_FILE_NOT_FOUND

```
public static final AECodes SETCERT_P12_FILE_NOT_FOUND
```

Не найден соответствующий файл p12 при установке сертификата в хранилище

CHANGEPASS_WRONG_PASSWORD

```
public static final AECodes CHANGEPASS_WRONG_PASSWORD
```

Неверный пароль при изменении пароля на хранилище

CHANGEPASS_EMPTY_NEWPASS

```
public static final AECodes CHANGEPASS_EMPTY_NEWPASS
```

Поле новый пароль пустое при изменении пароля на хранилище

CHANGEPASS_COMMON

```
public static final AECodes CHANGEPASS_COMMON
```

Общая ошибка при изменении пароля на хранилище

CHANGEPASS_INCORRECT_NEWPASS_PATTERN

```
public static final AECodes CHANGEPASS_INCORRECT_NEWPASS_PATTERN
```

Неверный формат нового пароля при изменении пароля на хранилище

CHANGEPASS_CONTAINS_EMPTY_ALIAS

```
public static final AECodes CHANGEPASS_CONTAINS_EMPTY_ALIAS
```

В хранилище не установлен сертификат при изменении пароля на хранилище

CHANGEPASS_NULL_KEYSTORE

```
public static final AECodes CHANGEPASS_NULL_KEYSTORE
```

Не удалось загрузить хранилище при изменении пароля на хранилище

SIGN_NULL_KEYSTORE

```
public static final AECodes SIGN_NULL_KEYSTORE
```

Не удалось загрузить хранилище при подписи

SIGN_EMPTY_STORAGE

```
public static final AECodes SIGN_EMPTY_STORAGE
```

Пустое хранилище при подписи

SIGN_EMPTY_PKCS12_STORAGE

```
public static final AECodes SIGN_EMPTY_PKCS12_STORAGE
```

В хранилище не установлен сертификат при подписи

SIGN_WRONG_PASSWORD

```
public static final AECodes SIGN_WRONG_PASSWORD
```

Неверный пароль при подписи

SIGN_COMMON

```
public static final AECodes SIGN_COMMON
```

Общая ошибка при подписи

SIGN_CANCEL

```
public static final AECodes SIGN_CANCEL
```

Отмена действия при подписи

TOKEN_CHOOSE_CANCEL

```
public static final AECodes TOKEN_CHOOSE_CANCEL
```

Отмена действия при выборе носителя

COMMON_CHOOSE_TOKEN_ERROR

```
public static final AECodes COMMON_CHOOSE_TOKEN_ERROR
```

Ошибка при выборе носителя

XML_PARSE_EXCEPTION

```
public static final AECodes XML_PARSE_EXCEPTION
```

Ошибка при чтении xml файла

BAD_XML_FORMAT

```
public static final AECodes BAD_XML_FORMAT
```

Неверный формат файла xml

SIGNATURE_ELEMENT_NOT_FOUND_WITHIN_XML

```
public static final AECodes SIGNATURE_ELEMENT_NOT_FOUND_WITHIN_XML
```

Элемент подписи (ds:Reference) не найден в структуре данного xml

EMPTY_SLOT

```
public static final AECodes EMPTY_SLOT
```

Устройства не найдены

UNKNOWN_SMARTCARD

```
public static final AECodes UNKNOWN_SMARTCARD
```

Неизвестное устройство

UNKNOWN_STORAGE

```
public static final AECodes UNKNOWN_STORAGE
```

Неизвестное хранилище

EXTENSIONS_AUTHORITY_INFORMATION_ACCESS_COMMON

```
public static final AECodes EXTENSIONS_AUTHORITY_INFORMATION_ACCESS_COMMON
```

Ошибка при генерации расширения: Доступ к сведениям центра сертификации

EXTENSIONS_AUTHORITY_KEY_IDENTIFIER_COMMON

```
public static final AECodes EXTENSIONS_AUTHORITY_KEY_IDENTIFIER_COMMON
```

Ошибка при генерации расширения: Идентификатор ключа центра сертификации

EXTENSIONS_BASIC_CONSTRAINTS_COMMON

```
public static final AECodes EXTENSIONS_BASIC_CONSTRAINTS_COMMON
```

Ошибка при генерации расширения: Базовые ограничения

EXTENSIONS_CRL_DISTRIBUTION_POINT_COMMON

```
public static final AECodes EXTENSIONS_CRL_DISTRIBUTION_POINT_COMMON
```

Ошибка при генерации расширения: Точки распределения списка отзыва

EXTENSIONS_CERTIFICATE_POLICY_COMMON

```
public static final AECodes EXTENSIONS_CERTIFICATE_POLICY_COMMON
```

Ошибка при генерации расширения: Политика сертификата

EXTENSIONS_EXTENDED_KEY_USAGE_COMMON

```
public static final AECodes EXTENSIONS_EXTENDED_KEY_USAGE_COMMON
```

Ошибка при генерации расширения: Расширенное использование ключа

EXTENSIONS_FRESHEST_CRL_COMMON

```
public static final AECodes EXTENSIONS_FRESHEST_CRL_COMMON
```

Ошибка при генерации расширения: Новейший CRL

EXTENSIONS_KNCA_ADMIN_COMMON

```
public static final AECodes EXTENSIONS_KNCA_ADMIN_COMMON
```

Ошибка при генерации расширения: Расширение определяющее администратора НУЦ РК

EXTENSIONS_KNCA_MANAGER_COMMON

```
public static final AECodes EXTENSIONS_KNCA_MANAGER_COMMON
```

Ошибка при генерации расширения: Расширение определяющее менеджера НУЦ РК

EXTENSIONS_KNCA_TRUSTED_OPERATOR_COMMON

```
public static final AECodes EXTENSIONS_KNCA_TRUSTED_OPERATOR_COMMON
```

Ошибка при генерации расширения: Расширение определяющее доверенного оператора НУЦ РК

EXTENSIONS_KEY_USAGE_COMMON

```
public static final AECodes EXTENSIONS_KEY_USAGE_COMMON
```

Ошибка при генерации расширения: Использование ключа

GENEXTENSION_CANCEL

```
public static final AECodes GENEXTENSION_CANCEL
```

Отмена действия при генерации расширения

LOAD_SLOT_LIST_EXCEPTION

```
public static final AECodes LOAD_SLOT_LIST_EXCEPTION
```

Ошибка при загрузке списка доступных устройств

RDN_NOT_FOUND

```
public static final AECodes RDN_NOT_FOUND
```

RDN не найден

EMPTY_KEY_LIST

```
public static final AECodes EMPTY_KEY_LIST
```

Пустой список ключей

GETPUBLICPART_EXCEPTION

```
public static final AECodes GETPUBLICPART_EXCEPTION
```

Ошибка при загрузке информации с удостоверения личности

SIGNATURE_VALIDATION_ERROR

```
public static final AECodes SIGNATURE_VALIDATION_ERROR
```

Ошибка проверки подписи

PRIVILEGED_ACTION_EXCEPTION

```
public static final AECodes PRIVILEGED_ACTION_EXCEPTION
```

Ошибка связанная с доступом

FILE_READ_ERROR

```
public static final AECodes FILE_READ_ERROR
```

Ошибка при чтении файла

Method Detail

values

```
public static AECodes[] values()
```

Returns an array containing the constants of this enum type, in the order they are declared. This method may be used to iterate over the constants as follows:

```
for (AECodes c : AECodes.values())
    System.out.println(c);
```

Returns:

an array containing the constants of this enum type, in the order they are declared

valueOf

```
public static AECodes valueOf(java.lang.String name)
```

Returns the enum constant of this type with the specified name. The string must match *exactly* an identifier used to declare an enum constant in this type. (Extraneous whitespace characters are not permitted.)

Parameters:

`name` - the name of the enum constant to be returned.

Returns:

the enum constant with the specified name

Throws:

`java.lang.IllegalArgumentException` - if this enum type has no constant with the specified name

`java.lang.NullPointerException` - if the argument is null