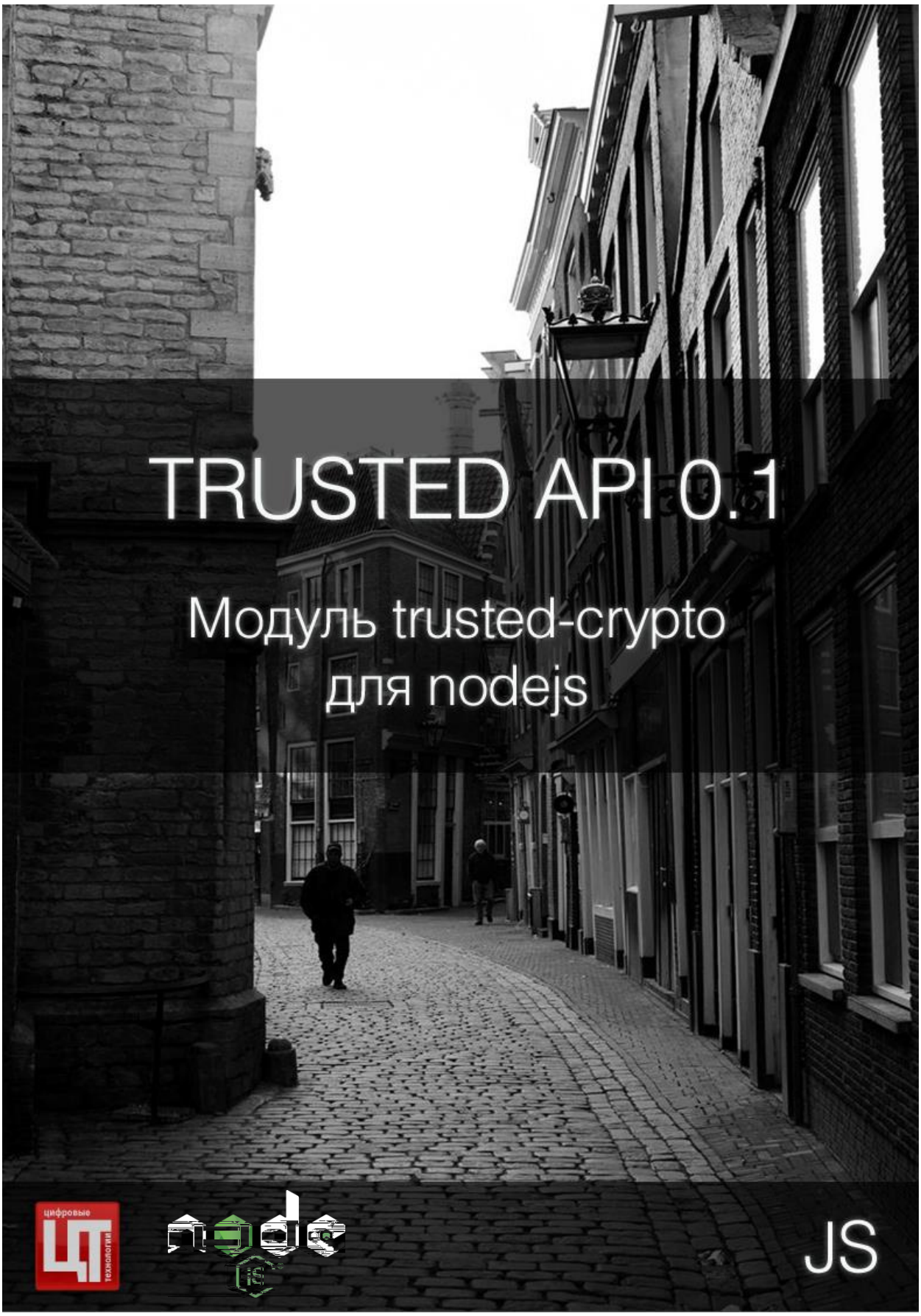
TRUSTED API 0.1
Модуль trusted-crypto
для nodejs
цифровые
технологии
JS

Оглавление

ktriCertCmp cert pki Certificate number

K CmsRecipientInfoCollection

push ri CmsRecipientInfo void

pop void

remove t index number void

Pki

K lgorithm

constructor

constructor handle native PKI lgorithm

constructor name string

duplicate lgorithm

isDigest boolean

K ttribute

dupicate ttribute

export Buffer

values ttributeValueCollection

values index number Buffer

K ttributeValueCollection

push val Buffer void

pop void

remove t index number void

items index number Buffer

K Certificate

compare cert Certificate number

equals cert Certificate boolean

hash algorithm string String

duplicate Certificate

load filename string format DataFormat void

static load filename string format DataFormat Certificate

import buffer Buffer format DataFormat void

static import buffer Buffer format DataFormat Certificate

export format DataFormat Buffer

save filename string format DataFormat void

K CertificationRequest

load filename string format DataFormat void

static load filename string format DataFormat CertificationRequest

sign key Key void

verify boolean

K CertificationRequestInfo

K CertificationCollection

items index number Certificate

push cert Certificate void

pop void

remove t index number void

K CertStore

addCertStore pvdType string pvdURI string void

removeCertStore pvdType string void

createCache cacheURI string void

addCacheSection cacheURI string pvdType string void

getPrvTypePresent pvdType string boolean

K Chain

buildChain cert Certificate certs CertificateCollection CertificateCollection

verifyChain chain CertificateCollection crls CrlCollection boolean

K Cipher

encrypt filenameSource string filenameEnc string format DataFormat void

decrypt filenameEnc string filenameDec string format DataFormat void

K Crl

getRevokedCertificateCert cer Certificate native PKI RevokedCertificate

getRevokedCertificateSerial serial string native PKI RevokedCertificate

load filename string format DataFormat void

static load filename string format DataFormat Crl

import buffer Buffer format DataFormat void

static import buffer Buffer format DataFormat Crl

export format DataFormat Buffer

save filename string dataFormat DataFormat void

compare crl Crl number

equals crl Crl boolean

hash algorithm string String

duplicate Crl

K CrlCollection

items index number Crl

push crl Crl void

pop void

remove t index number void

K Csr

save filename string dataFormat DataFormat void

K Key

generate format DataFormat pubExp PublicExponent keySize number password string Key

readPrivateKey filename string format DataFormat password string Key

writePrivateKey filename string format DataFormat password string any

readPublicKey filename string format DataFormat Key

writePublicKey filename string format DataFormat any

compare key Key number

K Oid

K Pkcs

certificate password string Certificate

key password string Key

ca password string CertificateCollection

load filename string void

static load filename string Pkcs

save filename string void

create cert Certificate key Key ca CertificateCollection password string name string Pkcs

K Revocation

getCrlLocal cert Certificate store PkiStore any

getCrlDistPoints cert Certificate rray string

checkCrlTime crl Crl boolean

downloadCRL distPoints rray string pathForSave string done Function void

PkiStore

K CashJson

export native PKISTORE IPkiItem

import items native PKISTORE IPkiItem void

K Filter

K PKIItem

K PkiStore

addProvider provider native PKISTORE Provider void

addCert provider native PKISTORE Provider category string cert Certificate string

addCrl provider native PKISTORE Provider category string crl Crl string

addKey provider native PKISTORE Provider key Key password string string

addCsr provider native PKISTORE Provider category string csr CertificationRequest string

find ifilter native PKISTORE IFilter native PKISTORE IPkiItem

findKey ifilter native PKISTORE IFilter native PKISTORE IPkiItem

getItem item native PKISTORE IPkiItem any

K Provider System

objectToPkiItem path string native PKISTORE IPkiItem

K ProviderCryptopro

K ProviderMicrosoft

П trusted crypto
Nodejs

Д
Nodejs У

С MS Visual Studio Windows g
Ubuntu XCode MacOS

З trusted crypto

Cms

SignedData

INTERF CES

	type SignedDataContentType data string Buffer

PROPERTIES

	ISignedDataContent	B
	rray string	B

METHODS

_____	B	true
_____	B	
_____	B	
_____ _____	B	
_____ _____ _____	B	
_____ _____	B	
_____ _____	B	
_____ _____ ___	C	
_____ _____	B	
_____	B	

isDetached boolean

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cms = trusted.cms.SignedData.load("test.sig", trusted.DataFormat.PEM);
console.log(cms.isDetached()); // false
```

certificates index number Certificate

	number И

ПРИМЕР

```
var trusted = require("trusted-crypto");
```

```
var cms = trusted.cms.SignedData.load("testsig.sig", trusted.DataFormat.PEM);
var certificate = cms.certificates(0);
console.log(certificate.signatureAlgorithm); // sha256
```

certificates CertificateCollection

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cms = trusted.cms.SignedData.load("testsig.sig", trusted.DataFormat.PEM);
var certificates = cms.certificates();
console.log(certificates.length); // 1
```

signers index number Signer

	number И

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cms = trusted.cms.SignedData.load("sigdoc.sig", trusted.DataFormat.PEM);
var signer = cms.signers(0);
console.log("Signer digest name:", signer.digestAlgorithm.name); // Signer digest name: sha1
```

signers SignerCollection

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cms = trusted.cms.SignedData.load("sigdoc.sig", trusted.DataFormat.PEM);
var signers = cms.signers();
for (var i = 0; i < signers.length; i++){
    var signer = signers.items(i);
    console.log("Signer digest name:", signer.digestAlgorithm.name); // Signer digest name:
sha1
```

load filename string format DataFormat void

	string	П				
	DataFormat	Ф	О	П		DER

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cms = trusted.cms.SignedData.load("testsig.sig", trusted.DataFormat.PEM);
```

static load filename string format DataFormat SignedData

	string	П			
	DataFormat	Ф	О	П	DER

import buffer Buffer format DataFormat void

	buffer	Б			
	DataFormat	Ф	О	П	DER

static import buffer Buffer format DataFormat SignedData

	buffer	Б			
	DataFormat	Ф	О	П	DER

export format DataFormat Buffer

	DataFormat	Ф	О	П	DER

save filename string format DataFormat void

	string	П			
	DataFormat	Ф	О	П	DER

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cms = trusted.cms.SignedData.load("testsig.sig", trusted.DataFormat.PEM);
cms.save("testsig1.sig", trusted.DataFormat.PEM);
```

createSigner cert Certificate key Key digestName string Signer

	Certificate	C
	Key	3

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cert = trusted.pki.Certificate.load("cert1.crt", trusted.DataFormat.PEM);
var key = trusted.pki.Key.readPrivateKey("cert1.key", trusted.DataFormat.PEM, "");
var sd = new trusted.cms.SignedData();
var signer = sd.createSigner(cert, key);
```

verify certs CertificateCollection boolean

	CertificateCollection	K

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cms = trusted.cms.SignedData.load("testsig.sig", trusted.DataFormat.PEM);
console.log(cms.verify()); //true
```

sign void

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cert = trusted.pki.Certificate.load("./test/cert.crt", trusted.DataFormat.PEM);
var key = trusted.pki.Key.readPrivateKey("./test/cert.key", trusted.DataFormat.PEM, "");
var sd = new trusted.cms.SignedData();
var signer = sd.createSigner(cert, key, "sha1");
  sd.content = {data: 'Hellow word'};
  sd.sign();
  sd.save("testsig.sig", trusted.DataFormat.PEM);
```

Vl qhu

PROPERTIES

	Certificate	3
	lgorithm	B
	SignerId	B

METHODS

	B
	B
	B
	B

signed ttributes Signer ttributeCollection

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cms = trusted.cms.SignedData.load("testsig.sig", trusted.DataFormat.PEM);
var signer = cms.signers(0);
var signedAttributes = signer.signedAttributes();
console.log(signer.digestAlgorithm.name);// sha1
console.log(signer.signedAttributes.length);//1
```

signed ttributes index number ttribute

	number	И

unsigned ttributes Signer ttributeCollection

unsigned ttributes index number ttribute

	number	И

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cms = trusted.cms.SignedData.load("testsig.sig", trusted.DataFormat.PEM);
```

```
var signer = cms.signers(0);
var unsignedAttributes = signer.unsignedAttributes();
console.log(unsignedAttributes.length);//-1
```

verifyContent v ISignedDataContent boolean

	any K

verify boolean

Vl qhuFrodf lrq

PROPERTIES

	number B

METHODS

	B

items index number Signer

	number И

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cms = trusted.cms.SignedData.load("testsig.sig", trusted.DataFormat.PEM);
var signers = cms.signers();
var signer = signers.items(0);
console.log(signers.length);// 1
console.log(signer.digestAlgorithm.name); //sha1
```

Vl qhuD ule hFrodhf lrq

PROPERTIES

	number B

METHODS

_____	B
_____	Д
_____	У

push attr ttribute void

	ttribute H

remove t index number void

	number И

items index number ttribute

	number И

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cms = trusted.cms.SignedData.load("testsig.sig", trusted.DataFormat.PEM);
var signer = cms.signers(0);
var signedAttributes = signer.signedAttributes();
console.log(signedAttributes.length);//-1
```

SignerId

PROPERTIES

	string B string B string B

ПРИМЕР

```
  var trusted = require("trusted-crypto");
  var cms = trusted.cms.SignedData.load("testsig.sig", trusted.DataFormat.PEM);
  var signer = cms.signers(0);
  var signerId = signer.signerId;
console.log(typeof signerId.issuerName); //string
  console.log(typeof signerId.serialNumber); // string
console.log(typeof signerId.keyId); //string
```

F p hfls lhq qir

PROPERTIES

	B
	B

METHODS

	B	CMS RecipientInfo

ktriCertCmp cert pki Certificate number

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cipher = new trusted.pki.Cipher();
var ris = cipher.getRecipientInfos("/encAssym.enc", trusted.DataFormat.PEM);
console.log(ris.length);
ri = ris.items(0);
console.log(ri.issuerName);
console.log(ri.serialNumber);
console.log(ri.ktriCertCmp(trusted.pki.Certificate.load("/cert1.crt",
trusted.DataFormat.PEM));
```

Fp hflslhq qirFrohf lrq

PROPERTIES

	B

METHODS

	Д у у

push ri CmsRecipientInfo void

pop void

remove t index number void

Pki

Algorithm

CONSTRUCTORS

	K K K

PROPERTIES

	string B Oid B

METHODS

	B B true

constructor

constructor handle native PKI lgorithm

constructor name string

duplicate lgorithm

isDigest boolean

ПРИМЕР

```
var trusted = require("trusted-crypto");
var alg = new trusted.pki.Algorithm('SHA');
console.log(alg.typeId.shortName); // SHA
console.log(alg.name);// sha
console.log(alg.duplicate().name); // sha
console.log(alg.isDigest()); //true
```

Attribute

CONSTRUCTORS

	К

PROPERTIES

	number	3	SN
	Oid	3	

METHODS

	В
	В DER
	В 3 DER

dupicate ttribute

export Buffer

values ttributeValueCollection

values index number Buffer

	number И

D ule h do hFr oohf lrq

CONSTRUCTORS

	К

PROPERTIES

	number В

METHODS

	Д у у в

push val Buffer void

	Buffer Н

pop void

remove t index number void

	number И

items index number Buffer

	number И

F hu lilf d h

CONSTRUCTORS

	K K

PROPERTIES

	number	B	
	string	B	
	number	B	
	number	B	KeyUsageFlags
	string	B	
	string	B	
	string	B	
	string	B	
	Date	B	
	Date	B	
	string	B	
	string	B	
	string	B	

METHODS

	C C B C Ч Ч C C

compare cert Certificate number

	Certificate C

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cert1 = trusted.pki.Certificate.load("test.crt");
var cert2 = trusted.pki.Certificate.load("test-ru.crt");
console.log (cert1.compare(cert2));  //  1
console.log(cert2.compare(cert1));   // -1
```

```
console.log(cert1.compare(cert1));    // 0
```

equals cert Certificate   boolean

	Certificate	C

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cert1 = trusted.pki.Certificate.load("test.crt");
var cert2 = trusted.pki.Certificate.load("test-ru.crt");
console.log(cert1.equals(cert2));       //false
console.log(cert1.equals(cert1));       // true
```

hash algorithm   string   String

	string	И	О	П	sha

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cert1 = trusted.pki.Certificate.load("test.crt");
console.log(cert1.hash());
```

duplicate   Certificate

	string	И	О	П	sha

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cert1 = trusted.pki.Certificate.load("test.crt");
var cert2 = cert1.duplicate();
console.log(cert1.thumbprint === cert2.thumbprint); // true
```

load filename string format   DataFormat   void

	string	П			
	DataFormat	Ф	О	П	DER

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cert = trusted.pki.Certificate.load("test.crt");
console.log(cert.serialNumber);
```

static load filename string format DataFormat Certificate

	string	П			
	DataFormat	Ф	О	П	DER

import buffer Buffer format DataFormat void

	Buffer	Б			
	DataFormat	Ф	О	П	DER

static import buffer Buffer format DataFormat Certificate

	Buffer	Б			
	DataFormat	Ф	О	П	DER

export format DataFormat Buffer

	DataFormat	Ф	О	П	DER

save filename string format DataFormat void

	string	П			
	DataFormat	Ф	О	П	DER

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cert = trusted.pki.Certificate.load("test.crt");
cert.save ('test_new.crt', trusted.DataFormat.PEM);
```

CertificationRequest

CONSTRUCTORS

	К К CertificationRequest

PROPERTIES

	Buffer П

METHODS

	Ч П П

load filename string format DataFormat void

	string	П			
	DataFormat	Ф	О	П	DER

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cr = new trusted.pki.CertificationRequest();
cr.load("test.csr");
console.log(cr.PEMString);
```

static load filename string format DataFormat CertificationRequest

	string	П			
	DataFormat	Ф	О	П	DER

sign key Key void

	Key K

verify boolean

Fhu lilfd lrq ht h qir

c

Fhu lilfd lrqFrohf lrq

CONSTRUCTORS

	К К CertificateCollection

PROPERTIES

	number В

METHODS

	В Д У У

items index number Certificate

	number И

ПРИМЕР

```
var trusted = require("trusted-crypto");
var certs = new trusted.pki.CertificateCollection();
certs.push(trusted.pki.Certificate.load("test.crt"));
var cert = certs.items(0);
console.log(cert.version); //2
```

push cert Certificate void

	Certificate

ПРИМЕР

```
var trusted = require("trusted-crypto");
var certs = new trusted.pki.CertificateCollection();
certs.push(trusted.pki.Certificate.load("test.crt"));
console.log(certs.length); //1
```

pop void

ПРИМЕР

```
var trusted = require("trusted-crypto");
var certs = new trusted.pki.CertificateCollection();
certs.push(trusted.pki.Certificate.load("test.crt"));
certs.pop();
console.log(certs.length); //0
```

---

remove t index number void

	number И

ПРИМЕР

```
var trusted = require("trusted-crypto");
var certs = new trusted.pki.CertificateCollection();
certs.push(trusted.pki.Certificate.load("test.crt"));
certs.push(trusted.pki.Certificate.load("test.crt"));
certs.removeAt(0);
console.log(certs.length); //1
```

F hu V r uh

CONSTRUCTORS

	К К CertStore

PROPERTIES

	string В

METHODS

	Д У С С П

addCertStore pvdType string pvdURI string void

	string Т string П

removeCertStore pvdType string void

	string Т

createCache cacheURI string void

	string П

addCacheSection cacheURI string pvdType string void

	string П string Т

getPrvTypePresent pvdType string boolean

	string	T

F kdlq

CONSTRUCTORS

	K

METHODS

	B B

buildChain cert Certificate certs CertificateCollection CertificateCollection

	Certificate	C
	CertificateCollection	K

ПРИМЕР

```
var trusted = require("trusted-crypto");
var  chain = new trusted.pki.Chain();
var certs = new trusted.pki.CertificateCollection();
var  cert1 = trusted.pki.Certificate.load("test.crt", trusted.DataFormat.DER);
    certs.push(cert1);
var  cert2 = trusted.pki.Certificate.load("cert.crt", trusted.DataFormat.PEM);
    certs.push(cert2);
cert3 = trusted.pki.Certificate.load("cert1.crt", trusted.DataFormat.PEM);
    certs.push(cert3);
var outchain = chain.buildChain(cert2, certs);
console.log(outchain.length);  //1
```

verifyChain chain CertificateCollection crls CrlCollection boolean

	CertificateCollection	K
	CrlCollection	K

F ls khu

CONSTRUCTORS

	К

PROPERTIES

	CryptoMethod	З
	CertificateCollection	З
	Key	З
	Certificate	З
	string	З
	string	З
	Buffer	З
	string	В
	Buffer	З
	string	В
	Buffer	З
	string	В
	String	В
	String	В
	String	В

METHODS

	Р
	Р
	В

encrypt filenameSource string filenameEnc string format DataFormat void

	string	И		
	String	З		
	DataFormat	Ф	П	DER

ПРИМЕР

```
var trusted = require("trusted-crypto");
cipher = new trusted.pki.Cipher("aes256");
cipher.cryptoMethod = trusted.CryptoMethod.SYMMETRIC;
cipher.digest = "MD5";
cipher.password = "4321";
cipher.encrypt("test.txt", "encSym.enc");
```

ПРИМЕР

```
var trusted = require("trusted-crypto");
var cipher = new trusted.pki.Cipher("aes256");
var cert = new trusted.pki.CertificateCollection();
cert.push(trusted.pki.Certificate.load("cert.crt", trusted.DataFormat.PEM));
cipher.recipientsCerts = cert;
cipher.encrypt("test.txt", "encAssym.enc", trusted.DataFormat.PEM);
```

decrypt filenameEnc string filenameDec string format DataFormat void

	string			
	string	P		
	DataFormat	Ф	П	DER

ПРИМЕР

```
var trusted = require('trusted-crypto');
var cipher = new trusted.pki.Cipher('aes256');
cipher.cryptoMethod = trusted.CryptoMethod.SYMMETRIC;
cipher.digest = "MD5";
cipher.password = "4321";
cipher.decrypt("encSym.enc", "decSym.txt", trusted.DataFormat.PEM);
```

F uo

CONSTRUCTORS

	K
	K CRL

PROPERTIES

	Buffer	B	SN	CRL	DER		
	Buffer	B					
	number	B					
	string	B					
	string	B					
	Date	B					
	Date	B					
	string	B		SH			
	string	B			CRL		
	string	B				CRL	
	string	B	OID		CRL		

METHODS

	crl

getRevokedCertificateCert cer Certificate native PKI RevokedCertificate

	Certificate C

ПРИМЕР

```
var trusted = require('trusted-crypto');
var crl = new trusted.pki.Crl();
var crl1 = trusted.pki.Crl.load("certcrl.crl");
var crl2 = crl1.getRevokedCertificateCert(trusted.pki.Certificate.load("test.crt"));
```

getRevokedCertificateSerial serial string native PKI RevokedCertificate

	string	C

load filename string format DataFormat void

	string	П			
	DataFormat	Ф	О	П	DER

ПРИМЕР

```
var trusted = require('trusted-crypto');
crl = new trusted.pki.Crl();
crl.load("certcrl.crl");
console.log(crl.sigAlgName);
```

static load filename string format DataFormat Crl

	string	П			
	DataFormat	Ф	О	П	DER

import buffer Buffer format DataFormat void

	Buffer	Б			
	DataFormat	Ф	О	П	DER

static import buffer Buffer format DataFormat Crl

	Buffer	Б			
	DataFormat	Ф	О	П	DER

export format DataFormat Buffer

	DataFormat	Ф	О	П	DER

save filename string dataFormat DataFormat void

	String	П				
	DataFormat	Ф	О	П	DER	

ПРИМЕР

```
var trusted = require('trusted-crypto');
var crl = new trusted.pki.Crl();
var crl1 = trusted.pki.Crl.load("certcrl.crl");
crl1.save ('certcrl2.crl');
```

compare crl Crl number

	Crl	Crl

ПРИМЕР

```
var trusted = require('trusted-crypto');
var crl = new trusted.pki.Crl();
var crl1 = trusted.pki.Crl.load("certcrl.crl");
var crl2 = trusted.pki.Crl.load("certcrl1.crl");
console.log(crl1.compare(crl2)); // 0
```

equals crl Crl boolean

	Crl	Crl

ПРИМЕР

```
var trusted = require('trusted-crypto');
var crl = new trusted.pki.Crl();
var crl1 = trusted.pki.Crl.load("certcrl.crl");
var crl2 = trusted.pki.Crl.load("certcrl1.crl");
console.log(crl1.equals(crl2)); // 0
```

hash algorithm string String

	string	H

ПРИМЕР

```
var trusted = require('trusted-crypto');
var Crl = new trusted.pki.Crl
var crl1 = trusted.pki.Crl.load("certcrl.crl");
var hash = crl1.hash("sha1");
console.log(hash);
```

duplicate Crl

ПРИМЕР

```
var trusted = require('trusted-crypto');
var Crl = new trusted.pki.Crl();
var crl1 = trusted.pki.Crl.load("certcrl.crl");
var crl2 = crl1.duplicate();
console.log(crl1.equals(crl2));
```

FuoFroohf lrq

CONSTRUCTORS

	К К CrlCollection

PROPERTIES

	number Р

METHODS

	В Д У у

items index number Crl

	number И

ПРИМЕР

```
var trusted = require('trusted-crypto');
var crls = new trusted.pki.CrlCollection();
crls.push(trusted.pki.Crl.load("certcrl.crl"));
var crl = crls.items(0);
console.log(crl.sigAlgName);
```

push crl Crl void

	Crl

ПРИМЕР

```
var trusted = require('trusted-crypto');
var crls = new trusted.pki.CrlCollection();
crls.push(trusted.pki.Crl.load("certcrl.crl"));
console.log(crls.length); //1
```

pop void

У

remove t index number void

	number И

ПРИМЕР

```
var trusted = require('trusted-crypto');
var crls = new trusted.pki.CrlCollection();
crls.push(trusted.pki.Crl.load("certcrl.crl"));
crls.removeAt(0);
console.log(crls.length); //0
```

F u

CONSTRUCTORS

PROPERTIES

	Buffer	П	Hex

METHODS

	C

save filename string dataFormat DataFormat void

	string	П				
	DataFormat	Ф	O	П		DER

ПРИМЕР

```
var trusted = require("trusted-crypto");
var key = trusted.pki.Key.readPrivateKey("cert.key", trusted.DataFormat.PEM, "");
var csr = new trusted.pki.CSR("/C=US/O=Test/CN=example.com", key, "SHA1");
csr.save("test.csr");
```

Key

CONSTRUCTORS

	К К Key

METHODS

	Г Ч З Ч З С

generate format DataFormat pubExp PublicExponent keySize number password string Key

	DataFormat PublicExponent number string	Ф Р П	О	П	DER

ПРИМЕР

```
var trusted = require('trusted-crypto');
var assert = require('assert');
var  key = new trusted.pki.Key();
var keyPair = key.generate(trusted.DataFormat.PEM, trusted.PublicExponent.RSA_F4, 1024);
assert.equal(keyPair === null, true, 'true');  // assert true
```

readPrivateKey filename string format DataFormat password string Key

	string DataFormat string	 Ф П	О	П	DER

ПРИМЕР

```
var trusted = require('trusted-crypto');
var  key = new trusted.pki.Key();
```

```
var privateKey = key.readPrivateKey("privkey_s.key", trusted.DataFormat.PEM, "1234");
assert.equal(privateKey == null, true, 'true');//true
```

writePrivateKey filename string format DataFormat password string any

	string				
	DataFormat	Ф	О	П	DER
	string	П			

ПРИМЕР

```
var trusted = require('trusted-crypto');
var key = new trusted.pki.Key();
var keyPair = key.generate(trusted.DataFormat.PEM, trusted.PublicExponent.RSA_F4, 1024);
keyPair.writePrivateKey("privkey_s.key", trusted.DataFormat.PEM, "1234");
```

readPublicKey filename string format DataFormat Key

	string				
	DataFormat	Ф	О	П	DER

ПРИМЕР

```
var trusted = require('trusted-crypto');
var assert = require('assert');
var key = new trusted.pki.Key();
publickey = key.readPublicKey("pubkey_s.key", trusted.DataFormat.PEM);
assert.equal(publickey == null, true, 'true'); // true
```

writePublicKey filename string format DataFormat any

	string				
	DataFormat	Ф	О	П	DER

ПРИМЕР

```
var trusted = require('trusted-crypto');
var key = new trusted.pki.Key();
var keyPair = key.generate(trusted.DataFormat.PEM, trusted.PublicExponent.RSA_F4, 1024);
keyPair.writePublicKey("pubkey_s.key", trusted.DataFormat.PEM);
```

compare key Key number

	Key	Key

ПРИМЕР

```
var trusted = require('trusted-crypto');
var  key = new trusted.pki.Key();
var  privateKey  =  key.readPrivateKey("privkey_s.key",  trusted.DataFormat.PEM,
"1234");
var  privateKey1  =  key.readPrivateKey("privkey_s.key",  trusted.DataFormat.PEM,
"1234");
console.log(privateKey.compare(privateKey1)); //1
```

Rlg

CONSTRUCTORS

	К
	К

Pkcs12

CONSTRUCTORS

	К К Pkcs

METHODS

	В В В

certificate password string Certificate

	string П

ПРИМЕР

```
var trusted = require('trusted-crypto');
var p12 = new trusted.pki.Pkcs12();
  p12.load("test.pfx");
var cert = p12.certificate("password");
```

key password string Key

	string П

ПРИМЕР

```
var trusted = require('trusted-crypto');
var p12 = new trusted.pki.Pkcs12();
  p12.load("test.pfx");
var key = p12.key("password");
```

ca password string CertificateCollection

	string П

ПРИМЕР

```
var trusted = require('trusted-crypto');
var p12 = new trusted.pki.Pkcs12();
  p12.load("test.pfx");
var ca = p12.ca("password");
console.log (ca.length);
```

load filename string void

	string	П

ПРИМЕР

```
var trusted = require('trusted-crypto');
var p12 = new trusted.pki.Pkcs12();
p12.load("test.pfx");
```

static load filename string Pkcs

	string	П

save filename string void

	string	П

ПРИМЕР

```
var trusted = require('trusted-crypto');
var p12 = new trusted.pki.Pkcs12();
var   cert = trusted.pki.Certificate.load("./test/cert.crt", trusted.DataFormat.PEM);
var   key = trusted.pki.Key.readPrivateKey("./test/cert.key", trusted.DataFormat.PEM, "");
var   p12Res = p12.create(cert, key, null, "1", "test_name");
p12Res.save('test_pkcs12.pfx');
```

create cert Certificate key Key ca CertificateCollection password string name string Pkcs

	Certificate	С
	Key	П
	CertificateCollection	Ц
	String	П
	string	П

ПРИМЕР

```
var trusted = require('trusted-crypto');
var p12 = new trusted.pki.Pkcs12();
var   cert = trusted.pki.Certificate.load("./test/cert.crt", trusted.DataFormat.PEM);
var   key = trusted.pki.Key.readPrivateKey("./test/cert.key", trusted.DataFormat.PEM, "");
var   p12Res = p12.create(cert, key, null, "1", "test_name");
```

Revocation

CONSTRUCTORS

	К

METHODS

	И crl В crl П crl З crl

getCrlLocal cert Certificate store PkiStore any

	Certificate PkiStore

getCrlDistPoints cert Certificate rray string

	Certificate C

ПРИМЕР

```
var trusted = require('trusted-crypto');
var revocation = new trusted.pki.Revocation();
var cert = trusted.pki.Certificate.load("test.crt");
var array = revocation.getCrlDistPoints(cert);
console.log(array);
```

checkCrlTime crl Crl boolean

	Crl CRL

downloadCRL distPoints rray string pathForSave string done Function void

	rray string T crl

	String	П
	Function	Ф

PkiStore

F d kMrq

CONSTRUCTORS

	K	JSON

METHODS

	B B

export native PKISTORE IPkiItem

ПРИМЕР

```
var trusted = require('trusted-crypto');
cashjson = new trusted.pkistore.CashJson('CertStore/cash.json');
var items = cashjson.export();
console.log(items.length);
```

import items native PKISTORE IPkiItem void

	native PKISTORE IPkiItem	M	PKI

ПРИМЕР

```
var trusted = require('trusted-crypto');
cashjson = new trusted.pkistore.CashJson('CertStore/cash.json');
var items = cashjson.export();
cashjson.import(items);
```

I lo hu

П PKI

CONSTRUCTORS

	K

PROPERTIES

	string	B	
	string	B	
	string	B	MY OTHER TRUST
	String	B	
	string	B	
	string	B	
	string	B	
	String	B	
	string	B	

S hp

CONSTRUCTORS

	K

PROPERTIES

	string	3		
	string	3		
	string	3		
	string	3		MY OTHER TRUST
	string	3	URI	
	string	3		
	string	3		
	string	3		
	string	3		
	string	3		
	string	3		
	String	3		
	string	3		
	string	3		
	string	3		
	string	3		
	boolean	3		
	string	3		
	string	3		

SnlV ruh

CONSTRUCTORS

	К К PkiStore

PROPERTIES

	CashJson	В	JSON

METHODS

	Д Д Д Д Д П П И

addProvider provider native PKISTORE Provider void

	native PKISTORE Provider	П

ПРИМЕР

```
var providerSystem = new trusted.pkistore.Provider_System('/CertStore');
store.addProvider(providerSystem.handle);
```

addCert provider native PKISTORE Provider category string cert Certificate string

	native PKISTORE Provider string Certificate	П К С	MY OTHER TRUST

ПРИМЕР

```
var trusted = require('trusted-crypto');
var providerSystem = new trusted.pkistore.Provider_System('/CertStore');
var store = new trusted.pkistore.PkiStore("/CertStore/cash.json");
var cert = trusted.pki.Certificate.load("cert1.crt", trusted.DataFormat.PEM);
store.addCert(providerSystem.handle, "MY", cert);
```

```
providerSystem = new trusted.pkistore.Provider_System('/CertStore');
store.addProvider(providerSystem.handle);
var items = store.find();
store.cash.import(items);
```

addCrl provider native PKISTORE Provider category string crl Crl string

	native PKISTORE Provider	П	
	string	К	MY OTHER TRUST
	Crl	С	

ПРИМЕР

```
var trusted = require('trusted-crypto');
var providerSystem = new trusted.pkistore.Provider_System('/CertStore');
var store = new trusted.pkistore.PkiStore("CertStore/cash.json");
crl = trusted.pki.Crl.load("certcrl.crl");
store.addCrl(providerSystem.handle, "CRL", crl);
providerSystem = new trusted.pkistore.Provider_System('/CertStore');
store.addProvider(providerSystem.handle);
var items = store.find();
store.cash.import(items);
```

addKey provider native PKISTORE Provider key Key password string string

	native PKISTORE Provider	П
	Key	К
	string	П

ПРИМЕР

```
var trusted = require('trusted-crypto');
var providerSystem = new trusted.pkistore.Provider_System('CertStore');
var store = new trusted.pkistore.PkiStore("CertStore/cash.json");
var  key = trusted.pki.Key.readPrivateKey("cert.key", trusted.DataFormat.PEM, "");
store.addKey(providerSystem.handle, key, "password");
providerSystem = new trusted.pkistore.Provider_System('/CertStore');
store.addProvider(providerSystem.handle);
var items = store.find();
store.cash.import(items);
```

addCsr provider native PKISTORE Provider category string csr CertificationRequest string

	native PKISTORE Provider	П	
	string	К	MY OTHER TRUST
	CertificationRequest	З	

ПРИМЕР

```
var trusted = require('trusted-crypto');
var providerSystem = new trusted.pkistore.Provider_System('/CertStore');
var store = new trusted.pkistore.PkiStore("/CertStore/cash.json");
var  csr = trusted.pki.CertificationRequest.load("test.csr", trusted.DataFormat.PEM, "");
store.addCsr(providerSystem.handle, "MY", csr);
providerSystem = new trusted.pkistore.Provider_System('/CertStore');
store.addProvider(providerSystem.handle);
var items = store.find();
store.cash.import(items);
```

find ifilter native PKISTORE IFilter native PKISTORE IPkiItem

native PKISTORE IFilter	Ф PKI

ПРИМЕР

```
var trusted = require('trusted-crypto');
var providerSystem = new trusted.pkistore.Provider_System('CertStore');
var store = new trusted.pkistore.PkiStore("CertStore/cash.json");
store.addProvider(providerSystem.handle);
var items = store.find({type: ["CERTIFICATE"], category: ["MY"]});
console.log(items.length);
```

findKey ifilter native PKISTORE IFilter native PKISTORE IPkiItem

native PKISTORE IFilter	Ф PKI

ПРИМЕР

```
var trusted = require('trusted-crypto');
var providerSystem = new trusted.pkistore.Provider_System('CertStore');
var store = new trusted.pkistore.PkiStore("CertStore/cash.json");
store.addProvider(providerSystem.handle);
var key = store.findKey({
        type: ["CERTIFICATE"],
        provider: ["SYSTEM"],
        category: ["MY"],
         hash: ['67cd1d796cfb42d00166737c6e16d596cf83695e']
        });
console.log(key.uri);
```

getItem item native PKISTORE IPkiItem any

	native PKISTORE IPkiItem O

ПРИМЕР

```
var trusted = require('trusted-crypto');
var providerSystem = new trusted.pkistore.Provider_System('CertStore');
var store = new trusted.pkistore.PkiStore("CertStore/cash.json");
store.addProvider(providerSystem.handle);
var certs = store.find({
        type: ["CERTIFICATE"],
        category: ["MY"]
    });
var item = certs[0];
var object = store.getItem(item);
console.log(object.type);
```

Provider_System

П

CONSTRUCTORS

	К

METHODS

	В	PKI

objectToPkiItem path string native PKISTORE IPkiItem

string	

Sur lghuFryptopro

П К П CSP

CONSTRUCTORS

	К

METHODS

	В К П

getKey cert pki Certificate

	Certificate C

ПРИМЕР

```
var trusted = require('trusted-crypto');

var providerCryptopro = new trusted.pkistore.ProviderCryptopro();
var key = providerCryptopro.getKey(cert);
```

ProviderMicrosoft

П Microsoft Windows

CONSTRUCTORS

	К

METHODS

	В К П

getKey cert pki Certificate

	Certificate C

ПРИМЕР

```
var cert =
var providerMicrosoft = new trusted.pkistore.ProviderMicrosoft();
var key = providerMicrosoft.getKey(cert);
```