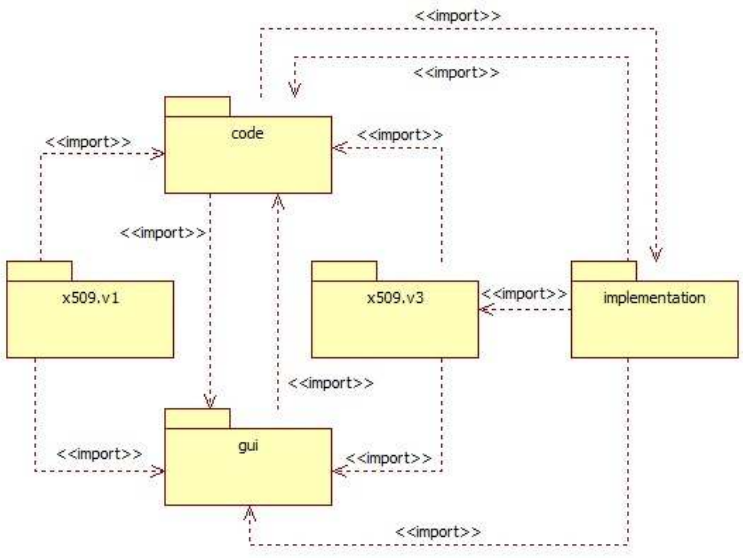
Priručnik za upotrebu priložene biblioteke

1. Struktura biblioteke

Izvorna biblioteka čijim korišćenjem studenti treba da dobiju aplikaciju koja ispunjava zadate zahteve raspoređena je u tri paketa: `gui`, `code` i jedan od `x509.v1` ili `x509.v3` (slika 1). Paketi `x509.v1` i `x509.v3` su osmišljeni tako da enkapsuliraju interfejs ka logici rada i interfejs ka GUI-u za određenu verziju X.509 digitalnih sertifikata. Da bi se dobila funkcionalna aplikacija potrebno je napraviti paket `implementation` i u njemu klasu `MyCode` koja nasleđuje klasu `CodeVX` iz paketa `x509.vX` (X označava zadatu verziju sertifikata - 1 ili 3). Ulazna tačka programa se nalazi u paketu `code` u klasi `X509`. Tu se instancira objekat klase `MyCode`. Konstruktor klase `MyCode` mora da ima sledeći potpis: `MyCode(boolean[] algorithm_conf, boolean[] extensions_conf, boolean extensions_rules) throws GuiException;` a parametri ovog konstruktora se prosleđuju konstruktoru osnovne klase `CodeVX`. Implementacijom nasleđenih apstraktnih metoda ispunjava se zadatak (slika 2). Klasa `CodeVX` sadrži i zaštićeno polje **`access`** tipa `GuiVX` koje predstavlja pristupnu tačku ka GUI-u i poseduje javne `get/set` metode za sva polja za unos i prikaz podataka na grafičkom korisničkom interfejsu. GUI je tako potpuno zatvoren za pristup osim preko tih javnih metoda klasa `GuiVX` i javno dostupnih konstantnih vrednosti koje se nalaze u okviru klase `Constants` u `gui` paketu. Time se ograničava skup operacija koje studenti mogu da izvedu sa grafičkim korisničkim interfejsom prilikom implementiranja logike aplikacije. Na ovaj način klasa `CodeVX` obuhvata i interfejs ka GUI-u i interfejs ka samoj implementaciji funkcionalnosti aplikacije.



Slika 1. Organizacija sistema po paketima

```
public interface CodeInterface {
    public abstract Enumeration<String> loadLocalKeystore();
    public abstract void resetLocalKeystore();
    // loadKey must return: -1 error, 0 not signed, 1 signed, 2 in case trusted certificate
    public abstract int loadKeypair(String keypair_name);
    public abstract boolean saveKeypair(String keypair_name);
    public abstract boolean removeKeypair(String keypair_name);
    public abstract boolean importKeypair(String keypair_name, String file, String password);
    public abstract boolean exportKeypair(String keypair_name, String file, String password);
    public abstract boolean importCertificate(String file, String keypair_name);
    public abstract boolean exportCertificate(String file, String keypair_name, int encoding, int format);
    public abstract boolean exportCSR(String file, String keypair_name, String algorithm);
    public abstract String importCSR(String file);
    public abstract boolean signCSR(String file, String keypair_name, String algorithm);
    public abstract boolean importCAReply(String file, String keypair_name);

    public abstract boolean canSign(String keypair_name);
    public abstract String getSubjectInfo(String keypair_name);
    public abstract String getCertPublicKeyAlgorithm(String keypair_name);
    public abstract String getCertPublicKeyParameter(String keypair_name);
}
```

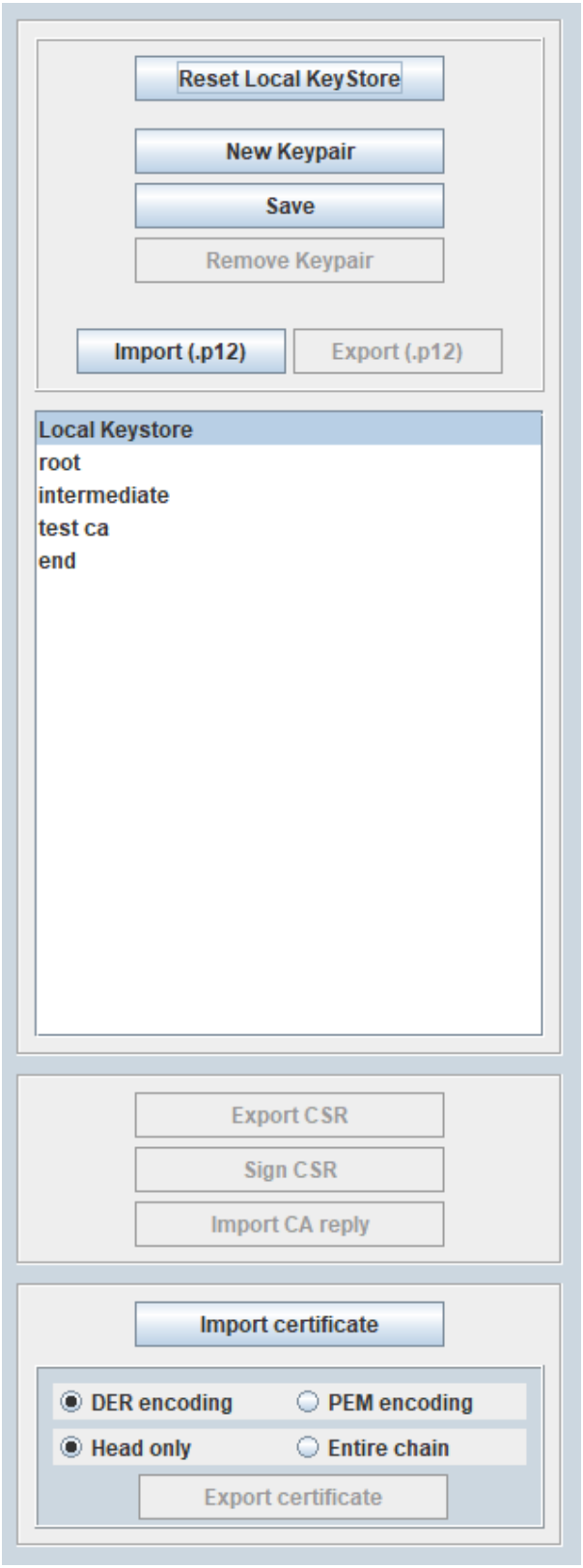
Slika 2. Metode klase `CodeVX` koje je potrebno implementirati

2. Objašnjenje metoda za implementaciju

Po pokretanju aplikacije na levoj strani prozora se nalazi *toolbar* panel (slika 3). Na njemu su postavljeni dugmići pomoću kojih se može upravljati sadržajem lokalnog *keystore*-a, kreirati i uvoziti/izvoziti parovi ključeva, potpisivati i uvoziti/izvoziti sertifikati i prikazivati detalji sertifikata na GUI-u, tako što se željeni sertifikat selektuje u listi koja oslikava sadržaj lokalnog *keystore*-a. Metode klase `CodeVX` koje je potrebno implementirati zapravo se pozivaju nakon što se proizvede određena akcija na *toolbar* panelu. Pritom se obavljaju sve provere korektnosti stanja grafičkog korisničkog interfejsa. Te provere o validnosti stanja GUI-a odnose se na provere da li su popunjena obavezna polja, da li su na mestima gde se očekuju brojevi vrednosti tipa **`int`**, proveru validnosti datuma, itd. U slučaju neke od tih grešaka korisnik se obaveštava prikazivanjem *pop-up* prozora sa informacijom o tipu greške. Ostale provere vezane za implementaciju rada sa X.509 sertifikatima, student je dužan sam da obezbedi. Postoje metode GUI interfejsa sa prijavu greške i prikazivanje informacija o njoj u *pop-up* prozoru. Sve informacije vezane za X.509 sertifikate mogu se pronaći na sledećim linkovima:

https://tools.ietf.org/html/rfc5280
https://tools.ietf.org/html/rfc2986
https://tools.ietf.org/html/rfc2315

Po pokretanju programa najpre se poziva `Enumeration<String> loadLocalKeystore();` metoda klase `CodeVX` koju je potrebno implementirati. Očekivana radnja koju treba da obavlja ta metoda je učitavanje liste svih *alias*-a parova ključeva/sertifikata iz lokalnog skladišta sertifikata u kolekciju tipa `Enumeration<String>`. Ako je metoda ispravno implementirana, u *Local Keystore* listi će se prikazati svi parovi ključeva/sertifikati koji su sačuvani u lokalnom skladištu ključeva. Ažuriranje sadržaja na ekranu je obezbeđeno grafičkim korisničkim interfejsom.



Slika 3. *Toolbar* panel

Pritiskom na dugme *Reset Local KeyStore* poziva se `void resetLocalKeystore();` metoda klase `CodeVX` koju je potrebno implementirati. Očekivana radnja koju treba da obavlja ta metoda je brisanje celokupnog sadržaja lokalnog skladišta sertifikata. Ako je metoda ispravno implementirana, *Local Keystore* lista će se isprazniti. Ažuriranje sadržaja na ekranu je obezbeđeno grafičkim korisničkim interfejsom.

Selektovanjem bilo kog para ključeva/sertifikata u listi *alias*-a *Local Keystore*-a poziva se `int loadKeypair(String keypair_name);` metoda klase `CodeVX` koju je potrebno implementirati. Ta metoda treba da učita podatke o paru ključeva/sertifikatu koji je sačuvan pod *alias*-om `keypair_name` iz lokalnog skladišta sertifikata i prikaže ih na grafičkom korisničkom interfejsu pomoću `get/set` metoda GUI interfejsa. Povratna vrednost metode je celobrojna vrednost koja označava uspešnost operacije (-1 u slučaju greške, 0 u slučaju da sertifikat sačuvan pod tim *alias*-om nije potpisan, 1 u slučaju da je potpisan, 2 u slučaju da je u pitanju *trusted* sertifikat). Prikazivanje podataka o paru ključeva na grafičkom korisničkom interfejsu obezbeđuje student pomoću `get/set` metoda polja **`access`**.

Pritiskom na dugme *New Keypair* resetuje se sadržaj polja za unos podataka o sertifikatu i pruža se mogućnost kreiranja novog para ključeva. Ažuriranje sadržaja na ekranu je obezbeđeno grafičkim korisničkim interfejsom.

Pritiskom na dugme *Save* prvo se vrši validacija svih polja na ekranu, a zatim se poziva `boolean saveKeypair(String keypair_name);` metoda klase `CodeVX` koju je potrebno implementirati. Očekivana radnja koju treba da obavlja ta metoda je generisanje i čuvanje novog para ključeva pod *alias*-om `keypair_name` u lokalno skladište sertifikata, na osnovu podataka sa GUI-a. Povratna vrednost metode označava uspešnost operacije (**`false`** u slučaju greške). Ako je metoda ispravno implementirana, u listi *alias*-a *Local Keystore*-a će se pojaviti *alias* sačuvanog para ključeva, a sadržaj polja za unos podataka će se resetovati. Ažuriranje sadržaja na ekranu i validacija polja obezbeđeni su grafičkim korisničkim interfejsom.

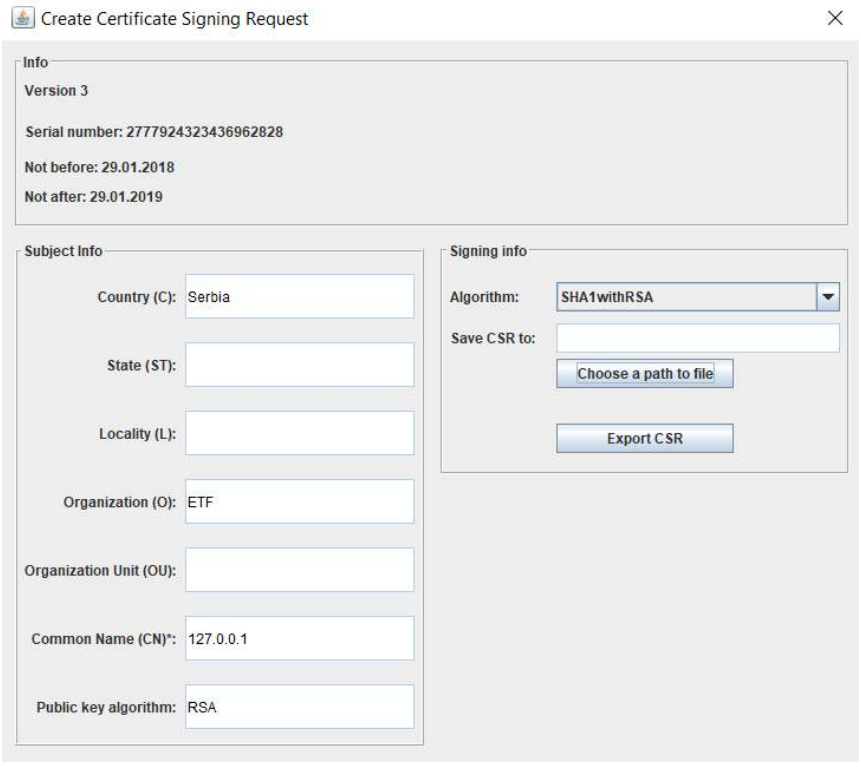
Pritiskom na dugme *Remove Keypair* poziva se `boolean removeKeypair(String keypair_name);` metoda klase `CodeVX` koju je potrebno implementirati. Ta metoda treba da ukloni par ključeva/sertifikat pod *alias*-om `keypair_name` iz lokalnog skladišta sertifikata. Povratna vrednost metode označava uspešnost operacije (**`false`** u slučaju greške). Ako je metoda ispravno implementirana, iz liste *alias*-a *Local Keystore*-a će nestati *alias* obrisanog para ključeva, a sadržaj polja za unos podataka će se resetovati. Ažuriranje sadržaja na ekranu je obezbeđeno grafičkim korisničkim interfejsom.

Pritiskom na dugme *Import (.p12)* poziva se `boolean importKeypair(String keypair_name, String file, String password);` metoda klase `CodeVX` koju je potrebno implementirati. Ta metoda treba da iz fajla sa putanjom `file` učita postojeći par ključeva koji je sačuvan u *PKCS#12* formatu i zaštićen lozinkom `password`, i sačuva ga u lokalno skladište sertifikata pod *alias*-om `keypair_name`. Povratna vrednost metode označava uspešnost operacije (**`false`** u slučaju greške). Ako je metoda ispravno implementirana, u listi *alias*-a *Local Keystore*-a će se pojaviti *alias* uvezenog para ključeva. Ažuriranje sadržaja na ekranu je obezbeđeno grafičkim korisničkim interfejsom.

Pritiskom na dugme *Export (.p12)* poziva se `boolean exportKeypair(String keypair_name, String file, String password)`; metoda klase `CodeVX` koju je potrebno implementirati. Ta metoda treba da selektovani postojeći par ključeva koji je u *Local Keystore*-u sačuvan pod *alias*-om `keypair_name` izveze, u *PKCS#12* formatu, u fajl sa putanjom `file` i zaštiti lozinkom `password`. Povratna vrednost metode označava uspešnost operacije (`false` u slučaju greške).

Pritiskom na dugme *Import certificate* poziva se `boolean importCertificate(String file, String keypair_name)`; metoda klase `CodeVX` koju je potrebno implementirati. Ta metoda treba da iz fajla sa putanjom `file` (ekstenzije .cer/.crt/.pem/.der) učita postojeći sertifikat i sačuva ga u lokalno skladište sertifikata pod *alias*-om `keypair_name`. Povratna vrednost metode označava uspešnost operacije (`false` u slučaju greške). Ako je metoda ispravno implementirana, u listi *alias*-a *Local Keystore*-a će se pojaviti *alias* uvezenog sertifikata. Ažuriranje sadržaja na ekranu je obezbeđeno grafičkim korisničkim interfejsom.

Pritiskom na dugme *Export certificate* poziva se `boolean exportCertificate(String file, String keypair_name, int encoding, int format)`; metoda klase `CodeVX` koju je potrebno implementirati. Ta metoda treba da postojeći sertifikat selektovan u *Local Keystore*-u pod *alias*-om `keypair_name` izveze u fajl sa putanjom `file` i kodira ga na način naznačen vrednošću parametra `encoding` (0 za DER, 1 za PEM). Parametar `int format` označava da li je pritom potrebno izvesti i ceo lanac sertifikata (0 za *Head only*, 1 za *Entire chain*). Povratna vrednost metode označava uspešnost operacije (`false` u slučaju greške).

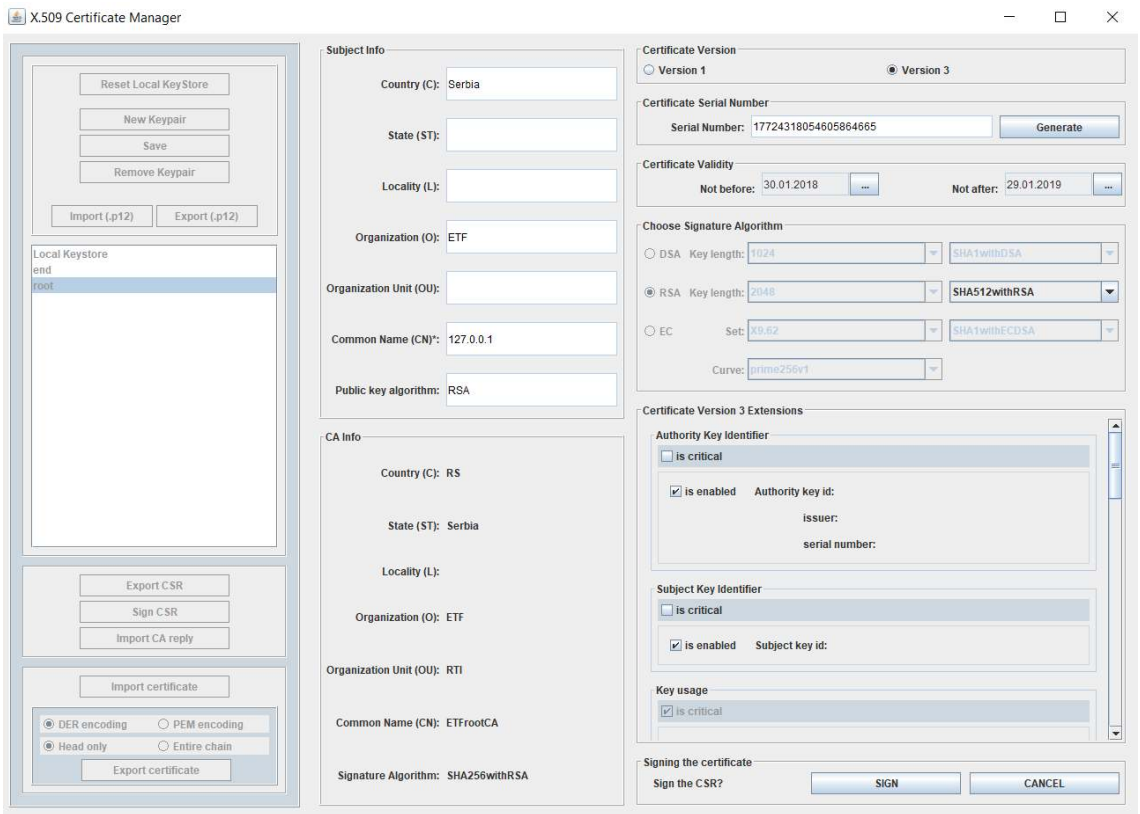


Slika 4. Dijalog za kreiranje *Certificate Signing Request*

Pritiskom na dugme *Export CSR* na *Toolbar* panelu otvara se novi dijalog za kreiranje zahteva za potpisivanje selektovanog sertifikata (*Certificate Signing Request*) koji je u *Local Keystor*e-u sačuvan pod *alias*-om `keypair_name` (slika 4).

Pritiskom na dugme *Export CSR* na dijalogu za kreiranje zahteva za potpisivanje poziva se `boolean exportCSR(String file, String keypair_name, String algorithm);` metoda klase `CodeVX` koju je potrebno implementirati. Ta metoda treba da generiše zahtev za potpisivanje selektovanog sertifikata (*Certificate Signing Request*) koji je u *Local Keystor*e-u sačuvan pod *alias*-om `keypair_name`, potpiše ga algoritmom `algorithm` i sačuva u fajlu sa putanjom `file` u *PKCS#10* formatu. Povratna vrednost metode označava uspešnost operacije (`false` u slučaju greške).

Pritiskom na dugme *Sign CSR* na na *Toolbar* panelu poziva se `String importCSR(String file);` metoda klase `CodeVX` koju je potrebno implementirati. Ta metoda treba da učita zahtev za potpisivanje selektovanog sertifikata (*Certificate Signing Request*) iz fajla sa putanjom `file`. Povratna vrednost metode je informacija o podnosiocu zahteva u formatu ATTRIBUTE=VALUE[,ATTRIBUTE=VALUE] gde ATTRIBUTE ima vrednost iz sledećeg skupa {C, S, L, O, OU, CN, SA}, a VALUE predstavlja vrednost tog atributa. Ako je metoda ispravno implementirana na ekranu će se prikazati informacije pročitane iz CSR-a i opcije za potpisivanje zahteva. Ažuriranje sadržaja na ekranu je obezbeđeno grafičkim korisničkim interfejsom.



Slika 5. Potpisivanje CSR-a

Pritiskom na dugme *Sign* poziva se **boolean** `signCSR(String file, String keypair_name, String algorithm);` metoda klase `CodeVX` koju je potrebno implementirati. Ta metoda treba da potpiše algoritmom `algorithm` sertifikat, za koji je kreiran CSR, privatnim ključem CA sertifikata koji je u lokalnom skladištu sertifikata sačuvan pod *alias*-om `keypair_name` i sačuva *CA reply* u *PKCS#7* formatu u fajl sa putanjom `file`. Povratna vrednost metode označava uspešnost operacije (**false** u slučaju greške). Podaci se prikupljaju sa GUI-a pomoću **access** polja.

Pritiskom na dugme *Import CA reply* poziva se **boolean** `importCAReply(String file, String keypair_name);` metoda klase `CodeVX` koju je potrebno implementirati. Ta metoda treba da iz fajla sa putanjom `file` učita CA *reply* za par ključeva koji je u lokalnom skladištu sertifikata sačuvan pod *alias*-om `keypair_name`. Povratna vrednost metode označava uspešnost operacije (**false** u slučaju greške). Ako je metoda ispravno implementirana na ekranu će se prikazati informacije o potpisanom sertifikatu. Ažuriranje sadržaja na ekranu je obezbeđeno grafičkim korisničkim interfejsom.

Pomoćna metoda `boolean canSign(String keypair_name);` klase `CodeVX` koju je potrebno implementirati treba da ispita da li je izabrani sertifikat koji je u lokalnom skladištu sertifikata sačuvan pod *alias*-om `keypair_name` sertifikacioni autoritet (CA) koji može da potpisuje druge sertifikate. Povratna vrednost **false** označava da sertifikat nije CA.

Pomoćna metoda `String getSubjectInfo(String keypair_name);` klase `CodeVX` koju je potrebno implementirati treba da vrati podatke o vlasniku izabranog sertifikata koji je u lokalnom skladištu sertifikata sačuvan pod *alias*-om `keypair_name`. Povratna vrednost treba da bude u formatu ATTRIBUTE=VALUE[,ATTRIBUTE=VALUE] gde ATTRIBUTE ima vrednost iz sledećeg skupa {C, S, L, O, OU, CN, SA}, a VALUE predstavlja vrednost tog atributa.

Pomoćna metoda `String getCertPublicKeyAlgorithm(String keypair_name);` klase `CodeVX` koju je potrebno implementirati treba da vrati podatke o algoritmu koji je korišćen za generisanje para ključeva izabranog sertifikata koji je u lokalnom skladištu sertifikata sačuvan pod *alias*-om `keypair_name`. Povratna vrednost treba da pripada sledećem skupu {DSA, RSA, EC}.

Pomoćna metoda `String getCertPublicKeyParameter(String keypair_name);` klase `CodeVX` koju je potrebno implementirati treba da vrati: dužinu ključa izabranog sertifikata koji je u lokalnom skladištu sertifikata sačuvan pod *alias*-om `keypair_name`, ako je algoritam RSA ili DSA; ime krive ako je algoritam EC.

3. Objašnjenje pristupanja GUI-u

Čitanje i prikazivanje podataka na grafičkom korisničkom interfejsu omogućeno je javnim metodama polja **access** u klasi `MyCode`. U tabeli ispod date su dostupne metode i njihova pojašnjenja.

Metoda GUI interfejsa	Komentar
`public static void reportError(Exception e);`	Prikazuje poruku (`e.getMessage()`) o izuzetku na GUI-u
`public static void reportError(String msg);`	Prikazuje `msg` poruku o greški na GUI-u
`public String getSelectedKeypair();`	Vraća vrednost para ključeva selektovanog u listi *Local Keystore* na *Toolbar* panelu
`public void addKeypair(String name);`	Dodaje `String name` u listu *Local Keystore* na *Toolbar* panelu
`public String getSubject();` `public void setSubject(String v);`	Vraća/postavlja vrednost svih *Subject input* polja u formatu ATTRIBUTE=VALUE[,ATTRIBUTE=VALUE] gde ATTRIBUTE ima vrednost iz sledećeg skupa {C, S, L, O, OU, CN}, a VALUE predstavlja vrednost tog atributa
`public String getSubjectCountry();` `public void setSubjectCountry(String v);`	Vraća/postavlja vrednost *input* polja *Subject Country*
`public String getSubjectState();` `public void setSubjectState(String v);`	Vraća/postavlja vrednost *input* polja *Subject State*
`public String getSubjectLocality();` `public void setSubjectLocality(String v);`	Vraća/postavlja vrednost *input* polja *Subject Locality*
`public String getSubjectOrganization();` `public void setSubjectOrganization(String v);`	Vraća/postavlja vrednost *input* polja *Subject Organization*
`public String getSubjectOrganizationUnit();` `public void setSubjectOrganizationUnit(String v);`	Vraća/postavlja vrednost *input* polja *Subject Organization Unit*
`public String getSubjectCommonName();` `public void setSubjectCommonName(String v);`	Vraća/postavlja vrednost *input* polja *Subject Common Name*
`public void setSubjectSignatureAlgorithm(String v);`	Postavlja vrednost *input* polja *Subject Signature Alorithm*
`public String getIssuer();` `public void setIssuer(String v);`	Vraća/postavlja vrednost svih *Issuer* labela u formatu

	ATTRIBUTE=VALUE[,ATTRIBUTE=VALUE] gde ATTRIBUTE ima vrednost iz sledećeg skupa {C, S, L, O, OU, CN}, a VALUE predstavlja vrednost tog atributa
`public String getIssuerSignatureAlgorithm();` `public void setIssuerSignatureAlgorithm(String v);`	Vraća/postavlja vrednost labele *Issuer Signature algorithm*
`public int getVersion();` `public void setVersion(int i);`	Vraća/postavlja vrednost radio dugmadi *Certificate Version*
`public String getSerialNumber();` `public void setSerialNumber(String v);`	Vraća/postavlja vrednost *input* polja *Certificate Serial Number*
`public Date getNotBefore();` `public void setNotBefore(Date d);`	Vraća/postavlja vrednost *input* polja *Validity Not before*
`public Date getNotAfter();` `public void setNotAfter(Date d);`	Vraća/postavlja vrednost *input* polja *Validity Not after*
`public String getPublicKeyAlgorithm();` `public void setPublicKeyAlgorithm(String v);`	Vraća/postavlja vrednost radio dugmadi *Certificate Public Key*. Parametar `String v` može imati vrednost iz skupa {DSA, RSA, EC}
`public String getPublicKeyParameter();` `public void setPublicKeyParameter(String v);`	Vraća/postavlja vrednost prve padajuće liste *Certificate Public Key Key Length* (ili *Set* u slučaju da je izabran EC algoritam)
`public String getPublicKeyDigestAlgorithm();` `public void setPublicKeyDigestAlgorithm(String v);`	Vraća/postavlja vrednost druge padajuće liste *Certificate Public Key Hash Algorithm*
`public String getPublicKeyECCurve();` `public void setPublicKeyECCurve(String v);`	Vraća/postavlja vrednost padajuće liste *Certificate Public Key Curve* (u slučaju da je izabran EC algoritam)

Samo za rad sa sertifikatima verzije 3	
`public boolean isSupported(int i);`	Ispituje da li je određena ekstenzija podržana konfiguracijom GUI-a
`public boolean isCritical(int i);`	Ispituje/postavlja vrednost *is critical*

`public void setCritical(int i, boolean v);`	*checkbox*-a za ekstenziju `int i`
`public String getPathLen();` `public void setPathLen(String v);`	Vraća/postavlja vrednost *input* polja *Path length* ekstenzije *Basic Constraints*
`public boolean isCA();` `public void setCA(boolean v);`	Ispituje/postavlja vrednost *is certificate authority checkbox*-a za ekstenziju *Basic Constraints*
`public String getPermittedNameConstraints();` `public void setPermittedNameConstraints(String v);`	Vraća/postavlja vrednost *input* polja *Permitted* ekstenzije *Name Constraints*
`public String getExcludedNameConstraints();` `public void setExcludedNameConstraints(String v);`	Vraća/postavlja vrednost *input* polja *Excluded* ekstenzije *Name Constraints*
`public boolean getEnabledAuthorityKeyID();` `public void setEnabledAuthorityKeyID(boolean v);`	Ispituje/postavlja vrednost *is enabled checkbox*-a za ekstenziju *Authority Key Identifier*
`public boolean getEnabledSubjectKeyID();` `public void setEnabledSubjectKeyID(boolean v);`	Ispituje/postavlja vrednost *is enabled checkbox*-a za ekstenziju *Subject Key Identifier*
`public void setAuthorityKeyID(String authorityKeyID);`	Postavlja vrednost *Authority key id* labele ekstenzije *Key Identifiers*
`public void setAuthorityIssuer(String authorityIssuer);`	Postavlja vrednost *Authority issuer* labele ekstenzije *Key Identifiers*
`public void setAuthoritySerialNumber(String authoritySerialNumber);`	Postavlja vrednost *Authority serial number* labele ekstenzije *Key Identifiers*
`public void setSubjectKeyID(String subjectKeyID);`	Postavlja vrednost *Subject key id* labele ekstenzije *Key Identifiers*
`public boolean [] getKeyUsage();` `public void setKeyUsage(boolean [] key_usage);`	Vraća/postavlja niz od 9 `boolean` vrednosti od kojih svaka označava vrednost jednog od *checkbox*-a za ekstenziju *Key Usage*
`public String getCpsUri();` `public void setCpsUri(String v);`	Vraća/postavlja vrednost *input* polja *CPS URI* ekstenzije *Certificate policies*

`public boolean getAnyPolicy();` `public void setAnyPolicy(boolean v);`	Ispituje/postavlja vrednost *any policy checkbox*-a ekstenzije *Certificate policies*
`public String [] getAlternativeName(int i);` `public void setAlternativeName(int i, String v);`	Vraća/postavlja vrednost *input* polja *Subject/Issuer alternative names,* ekstenzija *Subject/Issuer alternative name,* u zavisnosti od vrednosti parametra `int i.` Povratna vrednost `get` metode je niz stringova koji se dobije kada se vrednost *input* polja razdvoji ‘,’
`public String getSubjectDirectoryAttribute(int i);` `public void setSubjectDirectoryAttribute(int i, String v);`	Vraća/postavlja vrednost *input* polja *Place of birth,* odnosno *Country of citizenship,* ekstenzije *Subject directory attributes* u zavisnosti od vrednosti parametra `int i`
`public String getGender();` `public void setGender(String v);`	Vraća/postavlja vrednost *checkbox*-a *Gender* ekstenzije *Subject directory attributes.* Vrednosti su u skupu {M, F}
`public String getDateOfBirth();` `public void setDateOfBirth(String v);`	Vraća/postavlja vrednost *input* polja *Date of birth* ekstenzije *Subject directory attributes.* `String v` je formata “yyyyMMdd”
`public boolean [] getExtendedKeyUsage();` `public void setExtendedKeyUsage(boolean [] key_usage);`	Vraća/postavlja niz od 7 `boolean` vrednosti od kojih svaka označava vrednost jednog od *checkbox*-a za ekstenziju *Extended Key Usage*
`public String getSkipCerts();` `public void setSkipCerts(String v);`	Vraća/postavlja vrednost *input* polja *Skip certs* ekstenzije *Inhibit anyPolicy*
`public boolean getInhibitAnyPolicy();` `public void setInhibitAnyPolicy(boolean v);`	Ispituje/postavlja vrednost *inhibit anyPolicy checkbox*-a za ekstenziju *Inhibit anyPolicy*

Tabela 1. Metode GUI interfejsa

4. Konstante

Klasa `Constants` u paketu `gui` sadrži definisane javno dostupne konstante, koje se kao parametri prenose u metode GUI interfejsa na mestima gde je parametar naveden kao **`int`** `i` (slika 5). U tabeli 2 su navedene metode GUI interfejsa koje za parametre treba da prime neku vrednost klase `Constants`.

```
public class Constants {
    /* OBJECT IDENTIFIERS */
    public static final int NUM_OF_INFO = 7;
    public static final int C = 0,      // COUNTRY
                            ST = 1,     // STATE
                            L = 2,      // LOCALITY
                            O = 3,      // ORGANIZATION
                            OU = 4,     // ORGANIZATION UNIT
                            CN = 5,     // COMMON NAME
                            SA = 6,     // SIGNATURE ALGORITHM
                            UI = 7;     // UNIQUE IDENTIFIER
    /* X.509 VERSION NUMBER */
    public static final int V1 = 0, V2 = 1, V3 = 2;
    /* CERTIFICATE VALIDITY */
    public static final int NOT_BEFORE = 0, NOT_AFTER = 1;
    /* PUBLIC KEY ALGORITHMS */
    public static final int NUM_OF_ALGORITHMS = 4;
    public static final int DSA = 0, RSA = 1, GOST = 2, EC = 3;
    /* ENCODINGS */
    public static final int NUM_OF_ENCODINGS = 2;
    public static final int DER = 0, PEM = 1;
    public static final int NUM_OF_FORMATS = 2;
    public static final int HEAD = 0, CHAIN = 1;
    /* EXTENSIONS */
    public static final int NUM_OF_EXTENSIONS = 15;
    public static final int AKID = 0,       // AUTHORITY KEY IDENTIFIER
                            SKID = 1,       // SUBJECT KEY IDENTIFIER
                            KU = 2,         // KEY USAGE
                            CP = 3,         // CERTIFICATE POLICIES
                            PM = 4,         // POLICY MAPPINGS
                            SAN = 5,        // SUBJECT ALTERNATIVE NAME
                            IAN = 6,        // ISSUER ALTERNATIVE NAME
                            SDA = 7,        // SUBJECT DIRECTORY ATTRIBUTES
                            BC = 8,         // BASIC CONSTRAINTS
                            NC = 9,         // NAME CONSTRAINTS
                            PC = 10,        // POLICY CONSTRAINTS
                            EKU = 11,       // EXTENDED KEY USAGE
                            CRLDP = 12,     // CRL DISTRIBUTION POINTS
                            IAP = 13,       // INHIBIT ANYPOLICY
                            FCRL = 14;      // FRESHEST CRL
    public static final int NUM_OF_KU = 9;                  // NUMBER OF KEY USAGE OPTIONS
    public static final int KEY_CERT_SIGN = 5;              // INDEX OF CERTIFICATE SIGNING CHECKBOX
    public static final int NUM_OF_EKU = 7;                 // NUMBER OF EXTENDED KEY USAGE OPTIONS
    /* SUBJECT DIRECTORY ATTRIBUTES' OPTIONS */
    public static final int POB = 0,        // PLACE OF BIRTH
                            COC = 1,        // COUNTRY OF CITIZENSHIP
                            M = 0,          // MALE
                            F = 1;          // FEMALE
}
```

Slika 5. Konstante u klasi Constants

Metoda GUI interfejsa	Parametri
`public void setVersion(int i);`	`Constants.V1`, `Constants.V2`, `Constants.V3`
`public boolean isSupported(int i);`	`Constants.AKID`, `Constants.SKID`, `Constants.KU`, `Constants.CP`, `Constants.PM`, `Constants.SAN`, `Constants.IAN`, `Constants.SDA`, `Constants.BC`, `Constants.NC`, `Constants.PC`, `Constants.EKU`, `Constants.CRLDP`, `Constants.IAP`, `Constants.FCRL`
`public boolean isCritical(int i);`	
`public void setCritical(int i, boolean v);`	
`public String[] getAlternativeName(int i);`	`Constants.SAN`, `Constants.IAN`
`public void setAlternativeName(int i, String v);`	
`public String getSubjectDirectoryAttribute(int i);`	`Constants.POB`, `Constants.COC`
`public void setSubjectDirectoryAttribute(int i, String v);`	

Tabela 2. Konstante kao parametri metoda GUI interfejsa

.