ᘐᘣᘣᘐᘓᘖᘔᘐᘐ ᑲᓂᓭᘔᘐᓂᓀᑫᘣᘣᘐᘐ

ᘕᘔᘛᘣᘓᓭᘔᘐᘐ ᘓ ᘣᘐᘕᘓᘓᘛᘛᘔᘐᘐ ᑲᓂᓭᘔᘐᓂᓀᑫᘣᘣᘐᘐ

ᘣᓂᓀᘓᘐᘐ ᘛᘛᓂᘖᓂᓀ

Otvoreno računarstvo

Bežično i mobilno računarstvo

- Uvod - Web tehnologije na strani klijenta
- Svijet Jave - Appleti
- Svijet Jave - MIDleti
- WAP/WML - HTTP/HTML
- Java Card

Mario Žagar

Uvod

Bežično i mobilno računarstvo

- Komentar:
- bežičnu komunikaciju (*wireless*) mogu koristiti i fiksni i mobilni uređaji (računala)
  - različite kategorije bežičnih komunikacija i različita namjena
  - kod fiksnih uređaja, razlog korištenja je problem provlačenja “žica”
- mobilni uređaji (računala) funkcioniraju vrlo teško u žičanom okruženju
  - može se reći da mobilno računarstvo podrazumijeva bežične komunikacije, u nekim slučajevima to je osnova:
    - npr. bežične mreže osjetila (*Wireless Sensor Networks - WSN*)

Web vs. Internet

- Web je postao sinonim za Internet
  - Internet je globalna mreža orjentirana na prijenos podataka između krajnjih korisnika
  - Web je samo jedan od sustava koji koristi prijenos podataka putem Interneta, postoje i drugi servisi koji koriste Internetski prijenos podataka:
    - E-mail – asinkrona komunikacija
    - IM aplikacije – sinkrona komunikacija
    - VoIP – Internetska telefonija
    - FTP – slanje datoteka
    - Web usluge za slanje podataka

2 akronimi

- “B” za *business* (B2B)
- “C” za *consumer, client, citizen* (B2C)
- “G” za *government* (G2C, G2G, G2B)
  - Bilo što **s** “C” u sebi obično znači HTML formu (usluge preko Weba, korisničko sučelje je Web sučelje)
  - Bilo što **bez** “C” u sebi obično znači Web usluge (usluge preko Interneta, korisničko sučelje je Web usluga)

Napomena: da ne bi bilo zabune, “P” je iz sasvim druge priče (*Peer to peer*)

Web tehnologije na strani klijenta

- Uz “klasične preglednike” sve više do izražaja dolazi mobilnost
  - *“location based computing”*,*“context aware computing”*
  - mobilno računarstvo (*mobile computing*), sveprisutno računarstvo (*ubiquitous computing*), prožimajuće računarstvo (*pervasive computing*),....
- Različite poslovne aktivnosti
  - provjeravanje transakcija (kartično poslovanje)
- Usluge s različitim informacijama u stvarnom vremenu
  - npr. vremenska prognoza na mobitelu, pozicioniranje, ...
- Obično “surfanje” Internetom

Zahtjevi

- Višestruka uporaba
  - fleksibilnost, koliko god je to moguće
- Nezasvisnost o kontekstu uporabe
  - prenosivost, rad na različitim mobilnim platformama
- Uparivost s ostalim komponentama
  - zajednička sučelja
- Prikazivanje samo nužnih informacija
  - zbog veličine zaslona i njihove slabije rezolucije
- Norme

Bežično, mobilno računarstvo - naprave

- Naprave prilagođene korištenju od strane klijenata
  - Dlanovnik, *Personal Digital Assistant (PDA)*
  - Pametni telefon, *Smartphone*
  - *Gadget*
    - mala naprava za obavljanje nekog jednostavnog zadatka
      - zlobnici bi dodali "beskorisnog" :-)
- Dva pristupa:
  - aplikacije iz "velikih" sustava prilagođavaju se izvođenju na malim napravama (pojednostavljene inačice)
  - računalna snaga malih naprava raste, a time i mogućnost izvođenja složenih aplikacija

PDA – uvod

- Malo računalo s funkcijama mobitela
- iPAQ, BlackBerry, iPhone (?), ...
- Pristup Web uslugama (Internet, e-mail), pisanje i pregledavanje dokumenata, igrice...
- Touch-screen
- USB port, Ethernet port, WLAN, WiFi, kamera, proširiva memorija, Bluetooth, IrDA, GPS
- Sinkronizacija s PC-om
  - sinkronizacijska programska podrška na dlanovniku, npr. HotSync za Palm OS, ActiveSync za MS Windows

PDA – operacijski sustavi

- Palm OS (PalmSource) – 14,9% i pada
- iPhone (Apple) – 3% i raste
- Windows Mobile (Microsoft) – 49,2% i raste
- BlackBerry OS (Research In Motion) – 25 % i raste
- OpenZaurus, Familiar - različiti OS bazirani na Linuxu (free) – 0,7 %, stabilno
- Symbian OS – EPOC (Motorola, Panasonic, Nokia, Samsung, Siemens, SonyEricsson) – 5,8% i raste
- Ostali – 1,4%

Smartphone

- Mobilni telefoni s naprednim funkcijama
- Za razliku od PDA, naglasak je na telefonu s dodatnim funkcijama (računalo)
- Funkcije slične kao i na PDA
- Minijaturne QWERTY tipkovnice ili *touch-screen*
- Praotac NOKIA 9000 Communicator

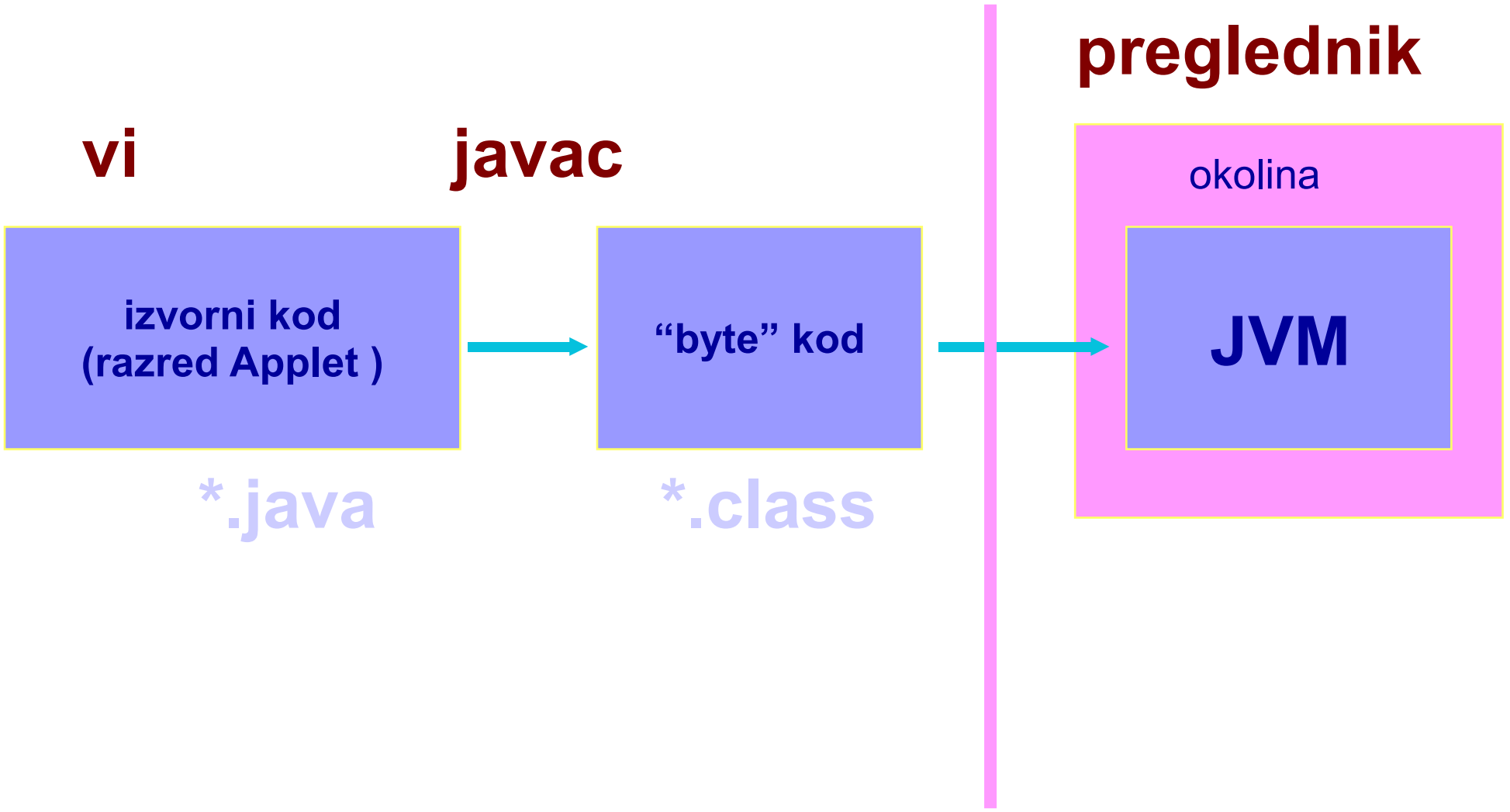
- *Smartphone - PDA*
  - isti cilj - pristup s različitih strana (telefonske i računalne)

Minijaturni Web preglednici

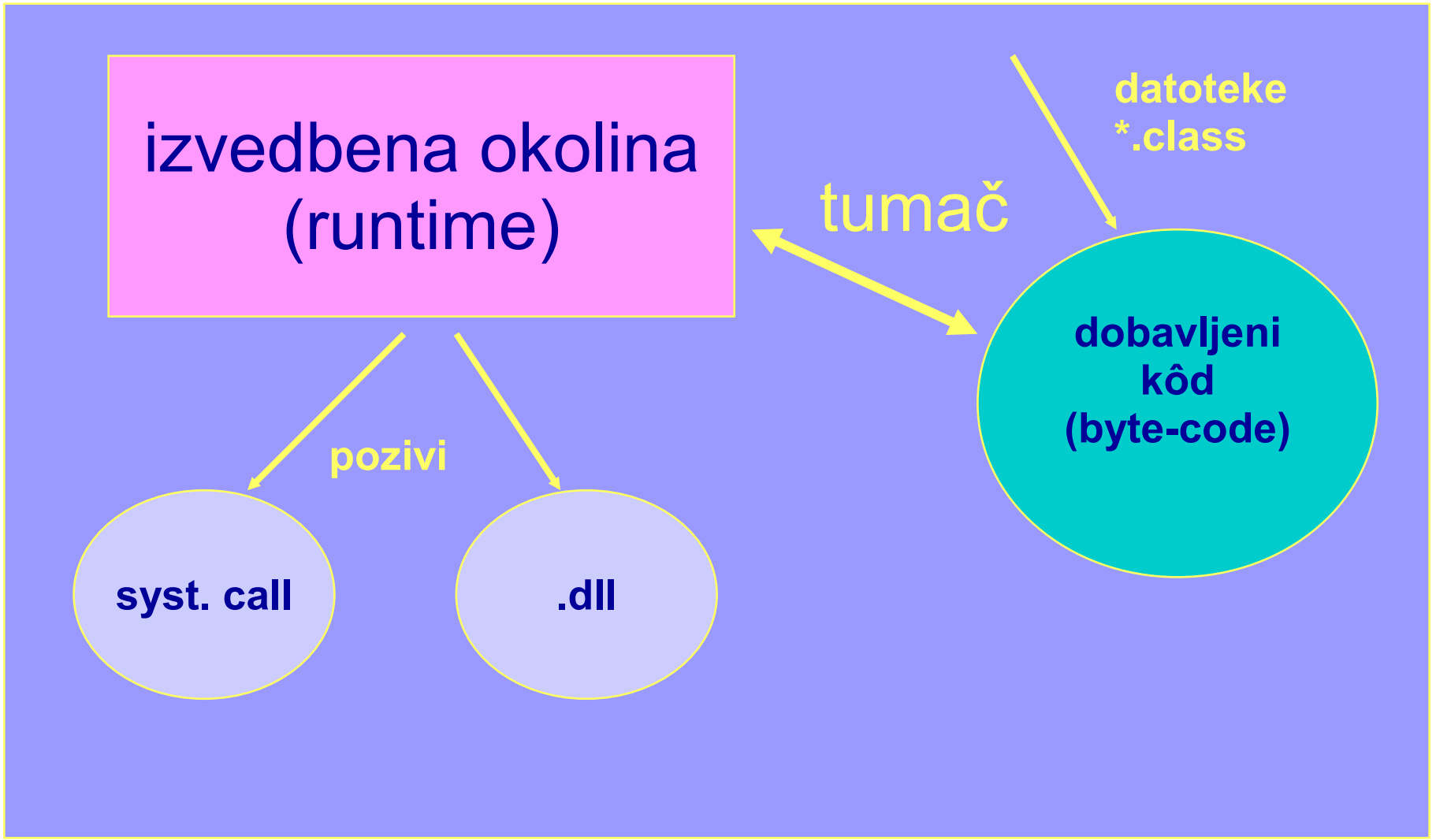
- Opera Mini
- Opera Mobile
- Safari
- NETFront

Svijet Jave - Appleti

Svijet Jave - Applet



Svijet Jave - Applet



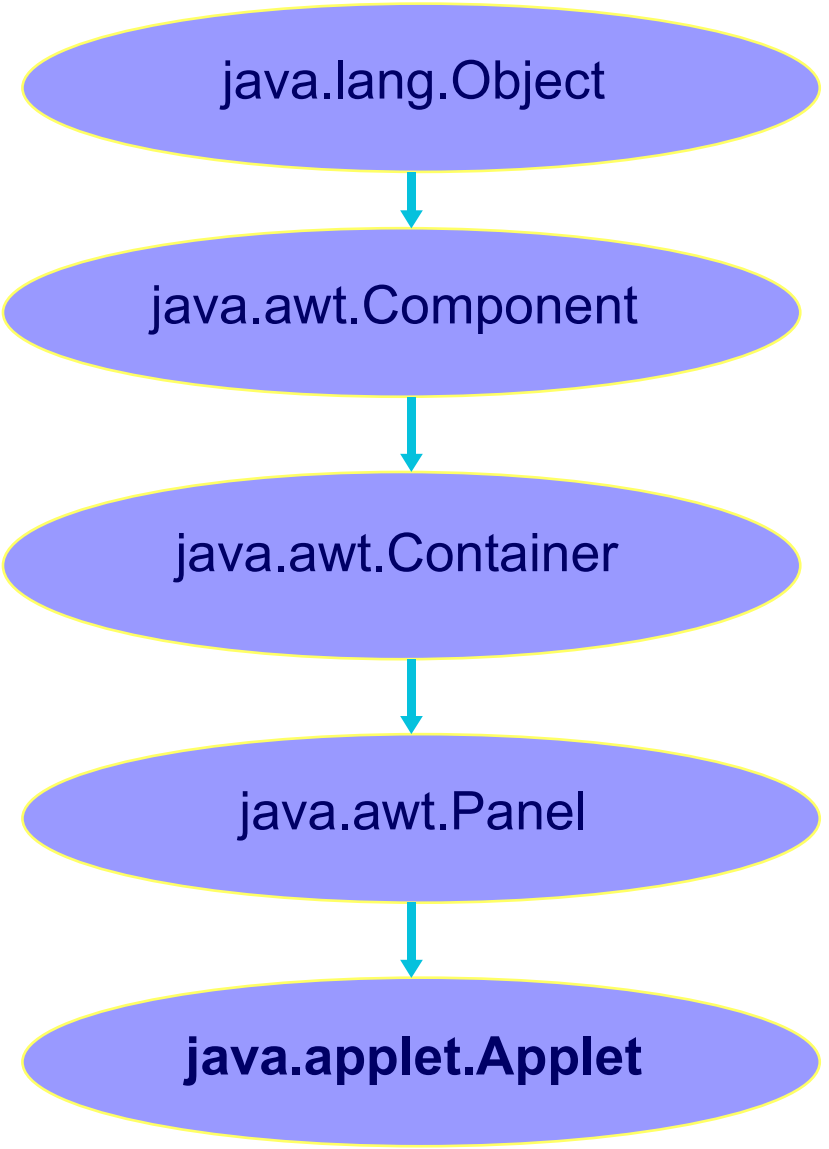
Poziv appleta

- izvođenje počinje prihvatom kôda preko stranice HTML (preglednik “otkrije” applet oznakom u HTML-u), npr:

```
<HTML>
<BODY>
<applet code="Applet3.class" width=300 height=200>
    <PARAM NAME=pozdrav VALUE="Srdačno vas pozdravljam">
    <PARAM NAME=kraj VALUE="i to bi bilo to!">
</applet>
<BODY>
</HTML>
```

Java applet

- Razlika između aplikacije i appleta
- Hijerarhija nasljeđivanja razreda Applet (**java.applet.Applet**)
- Time dobiva strukturu prikladnu izvođenju programa unutar preglednika
- Podrška ugrađena u preglednik



Životni vijek appleta

- metoda init()
  - izvodi se prva i postavlja grafičko korisničko sučelje, naš init nadjača onaj iz razreda Applet
- metoda start()
  - kada se pojavi stranica
- metoda stop()
  - kada se više ne prikazuje Web stranica
- metoda destroy()
  - izvodi se kad applet više nije potreban

Osnovne struktura appleta

```
public class Primjer extends Applet {
        . . .
        public void init() { . . . }
        public void start() { . . . }
        public void stop() { . . . }
        public void destroy() { . . . }
        . . .
    }
```

- a ako ima grafike, još: paint(), update(), repaint()
- nadjačavanje metoda (eng. overriding)

Applet - init()

- init - inicijalizacija appleta na početku (*load*) ili ponovljenom dohvatu (*reload*)
- Metoda init nema parametara i ne vraća nikakvu vrijednost, poželjno da sadrži kôd koji bi došao u konstruktor
- Razlog da appleti nemaju konstruktore je u tome da je okolina za izvođenje appleta kompletna tek nakon izvođenja metode *init()*
- U metodi *init()* poželjno je učitati slike, protumačiti parametare iz poziva appleta i sl.

Applet - start()

- *start* - pokretanje izvođenja appleta na početku ili kod ponovne posjete stranici
- Metoda start može se izvoditi više puta (za razliku od *init()* koji se izvodi samo jednom)
- Također može nadjačati metodu *start()* iz Appleta

Applet - paint()

- Nije/je standardna metoda(faza) u životu appleta
- Ispis na zaslon
- public void paint(Graphics g) {...}
- Parametar je instanca klase Graphics (objekt stvara preglednik)
- Potrebno import java.awt.Graphics

Appleti - crtanje

- Najjednostavniji applet prikazuje se na zaslonu nadvladavanjem (eng. overriding) metode paint

```
class Primjer extends Applet {
    . . .
    public void paint(Graphics g) { . . . }
    . . .
}
```

Metode za prikaz (crtanje):

- paint - osnovna metoda za prikaz, crtanje unutar preglednika
- update - metoda za poboljšanje crtanja (čisti zaslon i poziva paint)
- repaint - osvježava zaslon i pokreće (kao paralelna nit) update

Applet - stop()

- stop - zaustavljanje izvođenja appleta, prilikom napuštanja stranice ili kraja rada preglednika (suprotno od start)
- treba nadjačati

```
public void stop(){
    ....
}
```

- ako ne, nastavlja izvođenje u pozadini!

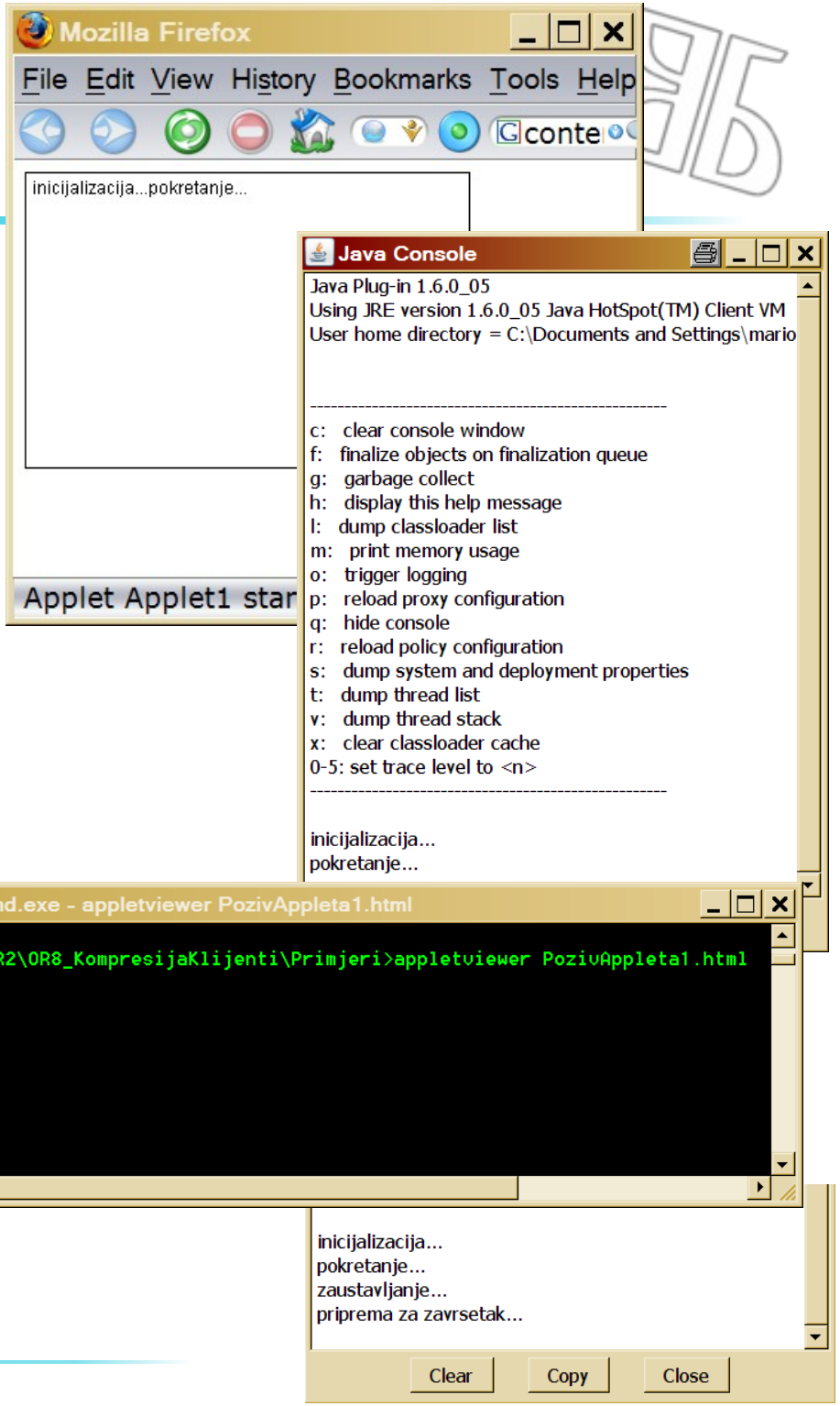
Applet - destroy()

- destroy - završno čišćenje i priprema za otpuštanje appleta (kraj rada, suprotno od init)
- samo jednom tijekom izvođenja appleta
- počisti za sobom (memoriju, suvišne objekte i dr.), treba nadjačati:

```
public void destroy(){
   ...
}
```

Primjer: Applet1.java

```
1  import java.applet.Applet;
2  import java.awt.Graphics;
3  public class Applet1 extends Applet {
4      StringBuffer spremnik;
5
6      public void init() {
7      spremnik = new StringBuffer();
8          addItem("inicijalizacija...");
9      }
10     public void start() {
11         addItem("pokretanje...");
12     }
13     public void stop() {
14         addItem("zaustavljanje...");
15     }
16     public void destroy() {
17         addItem("priprema za zavrsetak...");
18     }
19     void addItem(String novaRijec) {
20         System.out.println(novaRijec);
21         spremnik.append(novaRijec);
22         repaint();
23     }
24     public void paint(Graphics g) {
25     //pravokutnik kao rub
26         g.drawRect(0, 0,
27            getSize().width - 1,
28            getSize().height - 1);
29         g.drawString(spremnik.toString(), 5, 15);
30     }
31 }
```



Appleti - mogu:

- Appleti mogu “svirati”, “crtati”,...
- Povezati se sa posluživačem
- Prikazivati dokumente HTML
- Pozivati javne (*public*) metode drugih appleta unutar iste stranice
- Lokalni appleti nemaju restrikcije kao mrežni
- Izvoditi se i nakon napuštanja stranice

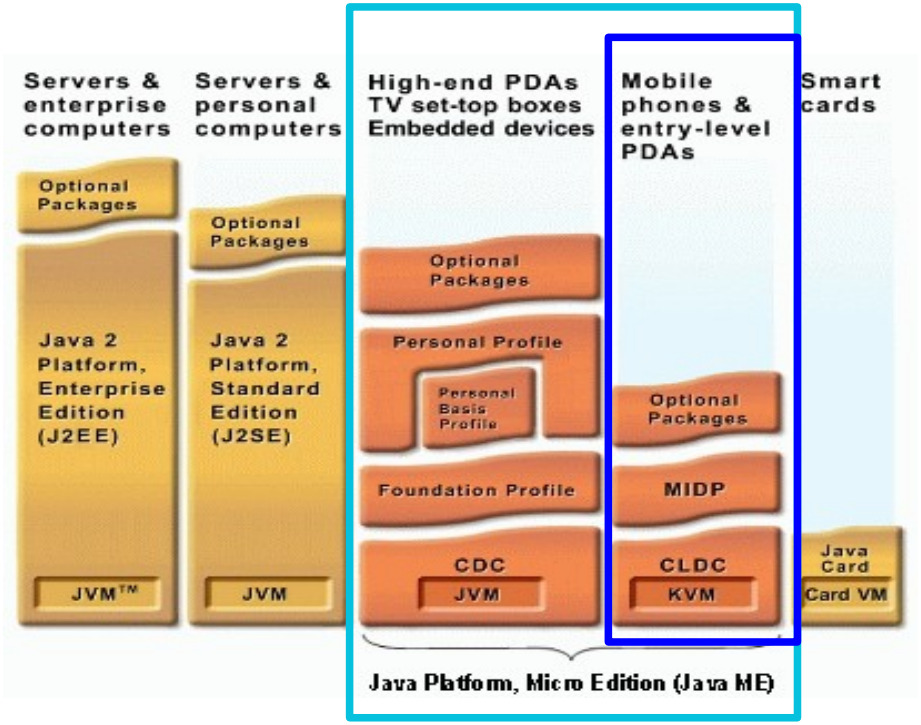
Appleti - NE mogu:

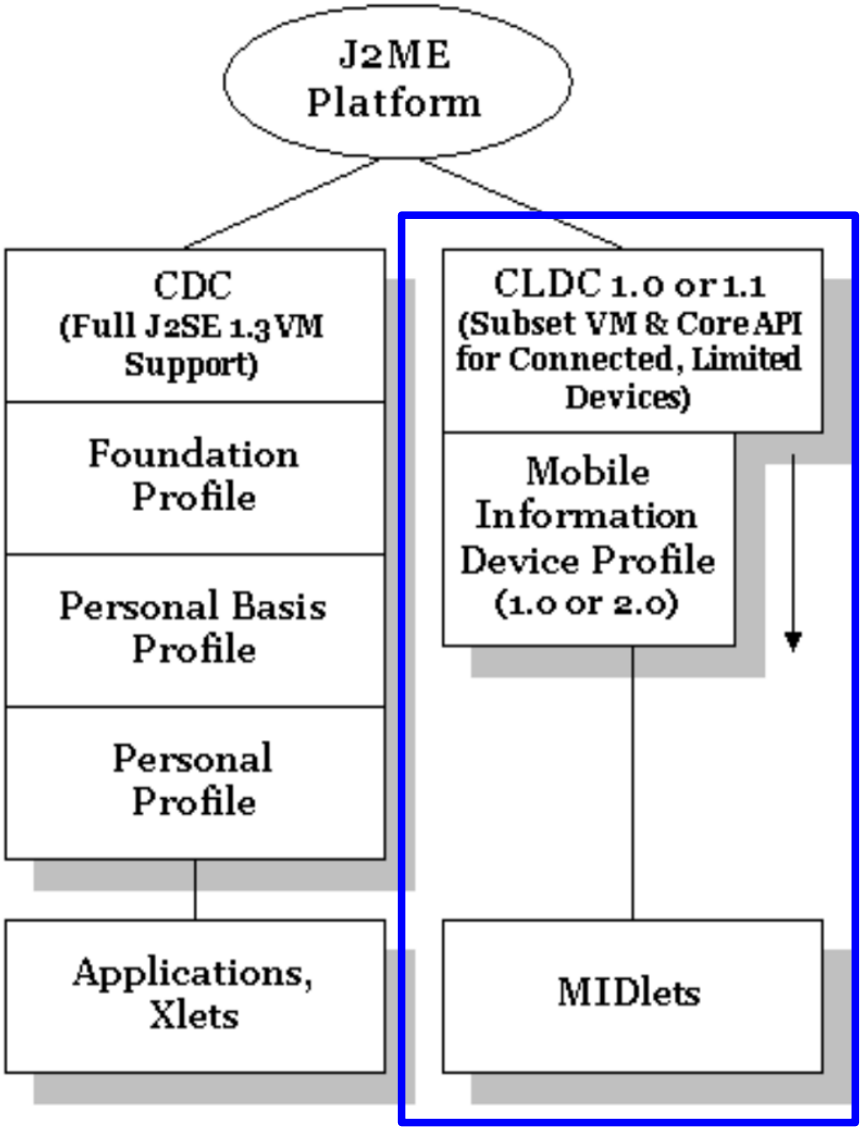
- Definirati *native* metode
- Napuniti (*loadati*) biblioteke
- Čitati/pisati u lokalne datoteke
- Ne mogu uspostavljati mrežnu komunikaciju osim s računalom iz kojeg su stigli
- Pokretati programe na računalu na kojem se nalaze

- Primjeri:

Svijet Jave - MIDleti

Java 2 Micro Edition (J2ME)





Java za male naprave (J2ME):

- Conected Limited Device Configuration (CLDC)
- Mobile Information Device Profile (MIDP)

CDC - CLDC

- CDC - uređaji s ograničenim resursima
  - *set-top-boxes*, navigacijski uređaji u automobilima,...
- CLDS - uređaji sa stvarno ograničenim resursima (memorija, procesorska snaga, komunikacijske mogućnosti,....)
  - PDA. mobiteli, *pageri*, ...
- Nema objektivnog kriterija da li je uređaj CDC ili CLDC, ovisi o proizvođaču
  - CLDC je podskup od CDC
  - CDC - potpuni JVM (inačica Compact VM - CVM, prenosiva i efikasna)

J2ME - *Profiles*

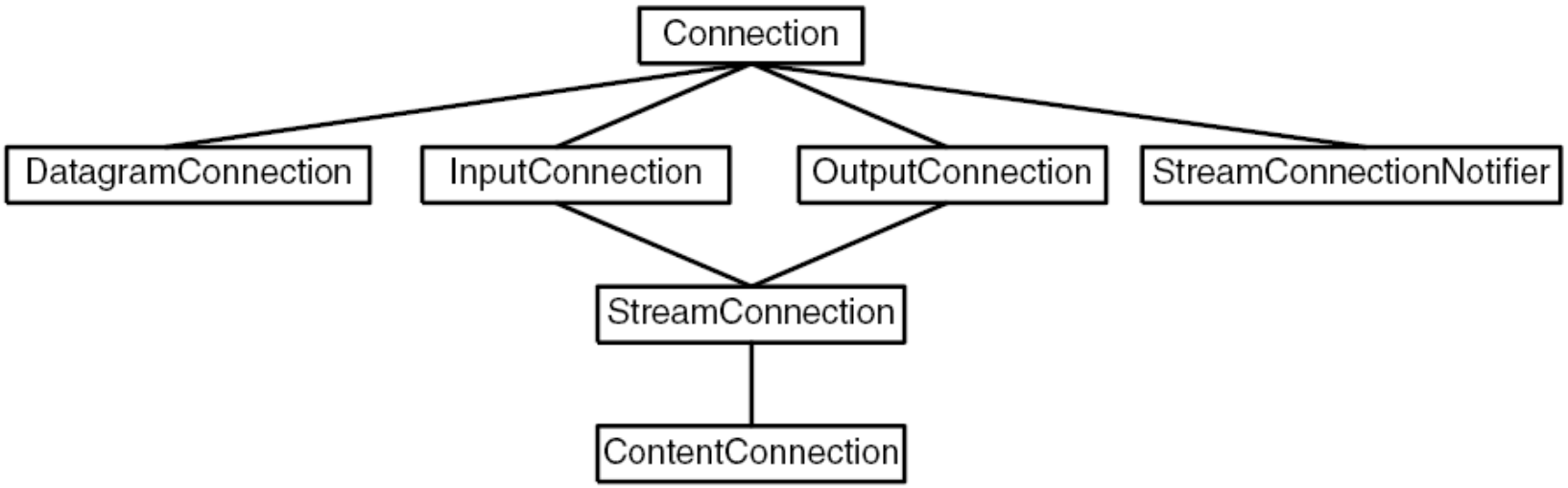
- Profili - podgrupe za određena područja korištenja
  - dvosjekli mač sa stanovišta prenosivosti
- *Personal Digital Assistant Profile (PDAP)*
  - proširuje funkcionalnost CLDC-a za PDA naprave
- *Foundation Profile*
  - dodaje neke razrede J2SE CDC-u, temelj za izgradnju drugih profila
- Personal Profile
  - proširuje *Foundation Profile*
- *RMI Profile*
  - dodaje R*emote Method Invocation* (RMI) CDC-u
- Mobile Information Device Profile (MIDP)

CLDC

- Najmanja J2ME implementacija
- Definirana kroz Java Community Process JSR-30 (http://java.sun.com/products/cldc)
- zahtjevi na resurse
  - 128K trajne memorije, 32K izbrisive memorije, može biti spora komunikacijska veza, mali kapacitet izvora napajanja (baterija), skromna procesorska snaga
- nema aritmetike pomičnog zareza
- nema finalizacije objekata (jednostavan rad sakupljača smeća)
- jednostavno rukovanje grješkama za vrijeme izvođenja
- podržan samo dio razreda iz java.lang, java.io, java.util
- nema java.awt, java.net, java.sql,.....

CLDC

- Uvodi se Generic Connection Framework (GCF)
  - skup apstraktnih razreda koji omogućuju stvaranje jednostavne i efikasne komunikacije, rada s datotekama i dr.

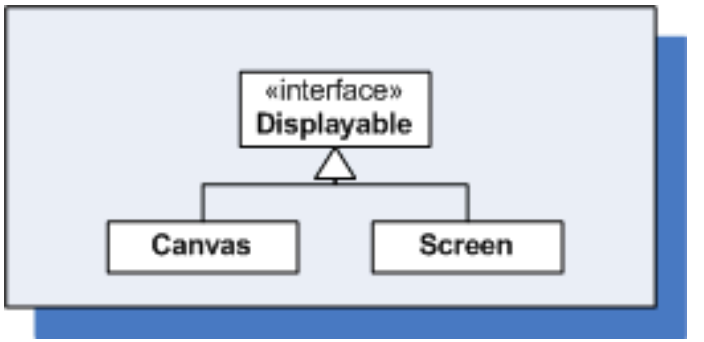


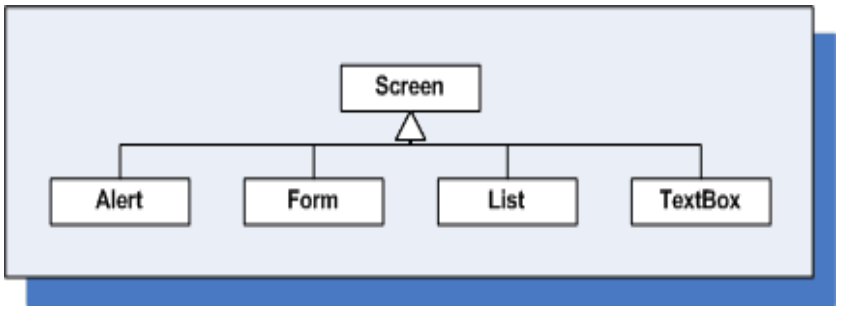
K Virtual Machine (KVM)

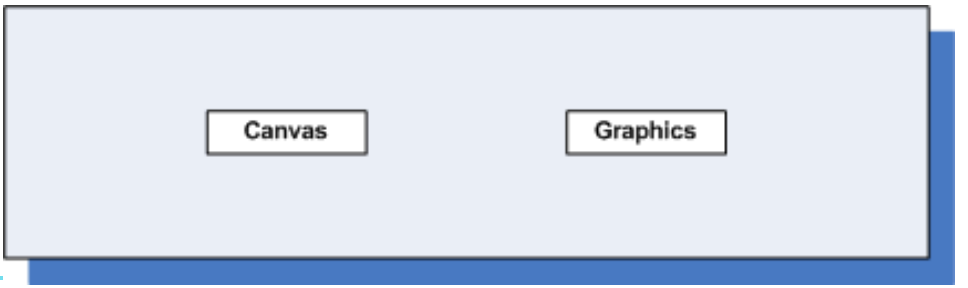
- KVM
  - podskup JVM-a
- Kompletno napisan u C-u s posebnim naglaskom na
  - optimiziranje za male sustave
  - prenosivost na različite platforme
  - modularnost i proširljivost
  - maksimalno optimiran kôd

MIDP

- *MIDP profile (danas 2.0)*
  - podrška mobitelima i sličnim napravama ograničenih resursa (veličina zaslona, tipkovnica, ograničena baterija, procesorska snaga, mrežna propusnost...)
  - niz API-a prilagođenih resursima, npr.:
    - import javax.microedition.midlet.*;
    - import javax.microedition.lcdui.*; // LCD User Interface
- MIDP aplikacija - skraćeni naziv MIDLet
- MIDLet sličan appletu ali se različito ponaša i ima različite zahtjeve







Primjer: MIDlet kMid

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;

public class kMid extends MIDlet implements CommandListener {

    kCanvas myCanvas;

    private Command izlazCommand = new Command("Izlaz", Command.EXIT, 99);
    private Command pokreniCommand =
                            new Command("Pokreni/Stani",Command.SCREEN, 1);

    public kMid() {
        myCanvas = new kCanvas();
        myCanvas.addCommand(izlazCommand);
        myCanvas.addCommand(pokreniCommand);
        myCanvas.setCommandListener(this);
        }
    public void startApp() throws MIDletStateChangeException {
        Display.getDisplay(this).setCurrent(myCanvas);
        myCanvas.repaint();
        }
```

Primjer: MIDlet kMid - nastavak

```
                                        ....
    public void destroyApp(boolean unconditional)
                                throws MIDletStateChangeException {
  }
    public void pauseApp() {
  }
    public void commandAction(Command c, Displayable s) {

        if(c==pokreniCommand) {
            myCanvas.pokreni();
            }
        else if (c==izlazCommand){
            try {
                destroyApp(false);
                notifyDestroyed();
            } catch (MIDletStateChangeException e) {
                e.printStackTrace();
            }
        }
    }
}
```

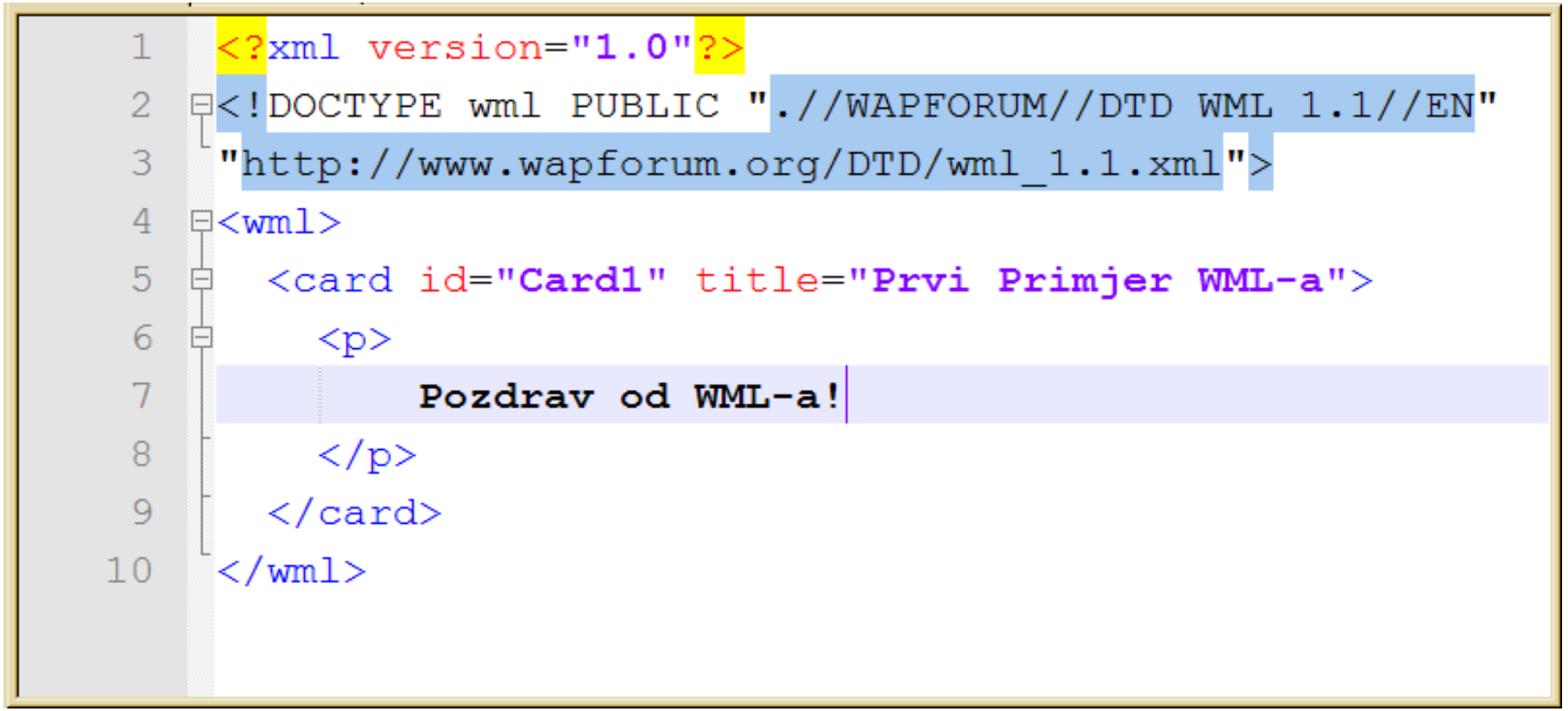
Primjer: MIDlet kMid - izlaz

WAP/WML - HTTP/XHTML

U početku bijaše WAP...

- WAP (*Wireless Application Protocol*)
  - norma za aplikacije koje koriste bežičnu komunikaciju
- Osnovna uporaba
  - pristup Internetu s mobilnog telefona ili PDA
- WAP 1.0 je bio reklamiran kao "*Internet u vašem džepu!*"
- WML - uglavnom tekst (bez tablica i slika)
- WML je XML aplikacija
- Stranice WML zovu se *DECKS*
  - podijeljene u *CARDS* međusobno povezane linkovima
  - navigacija unutar kartica (CARDS) je lokalna (na mobitelu)

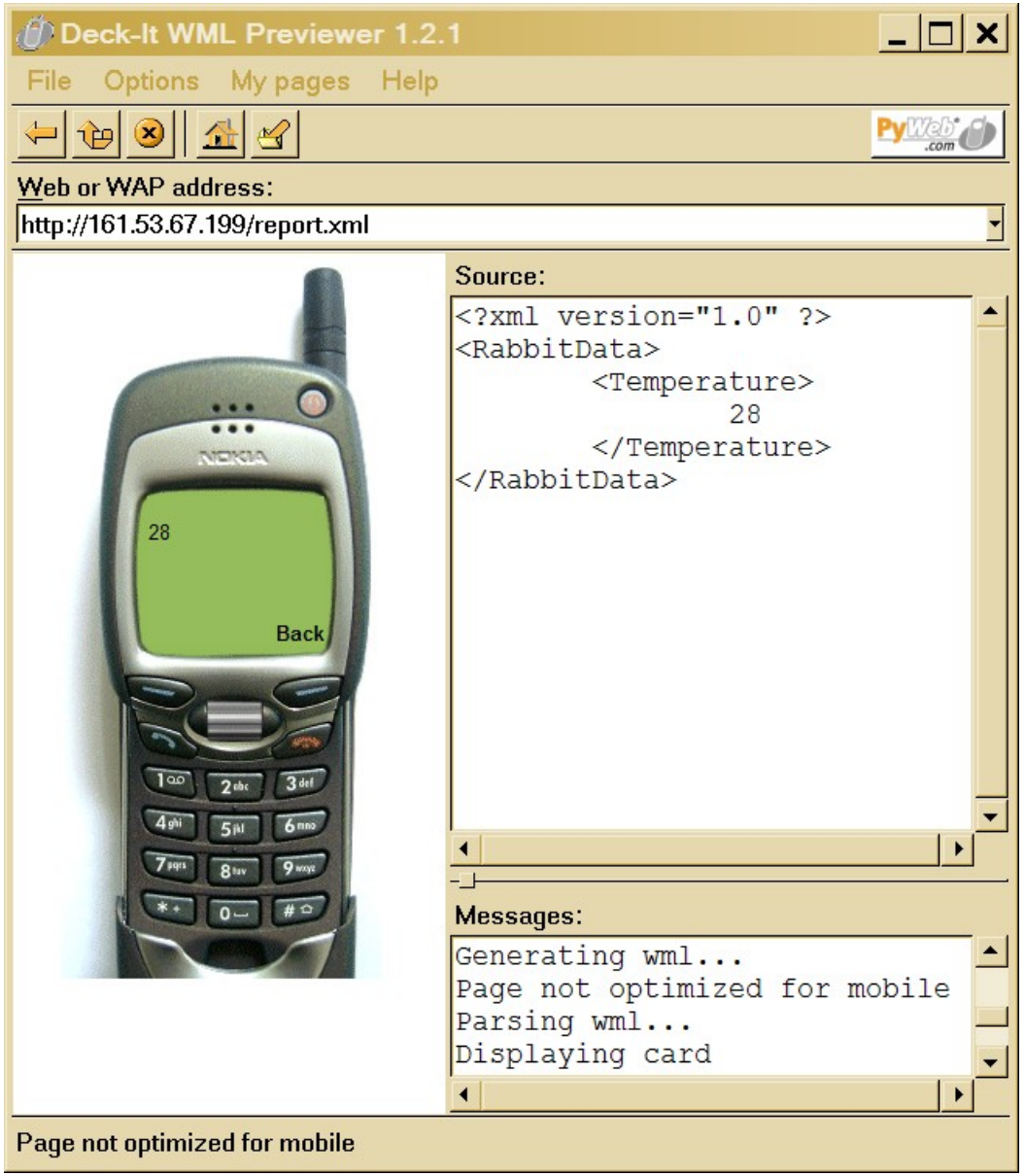
Pr1.wml



```
1  <?xml version="1.0"?>
2  <!DOCTYPE wml PUBLIC ".//WAPFORUM//DTD WML 1.1//EN"
3  "http://www.wapforum.org/DTD/wml_1.1.xml">
4  <wml>
5    <card id="Card1" title="Prvi Primjer WML-a">
6      <p>
7          Pozdrav od WML-a!
8      </p>
9    </card>
10 </wml>
```

Pr2.wml

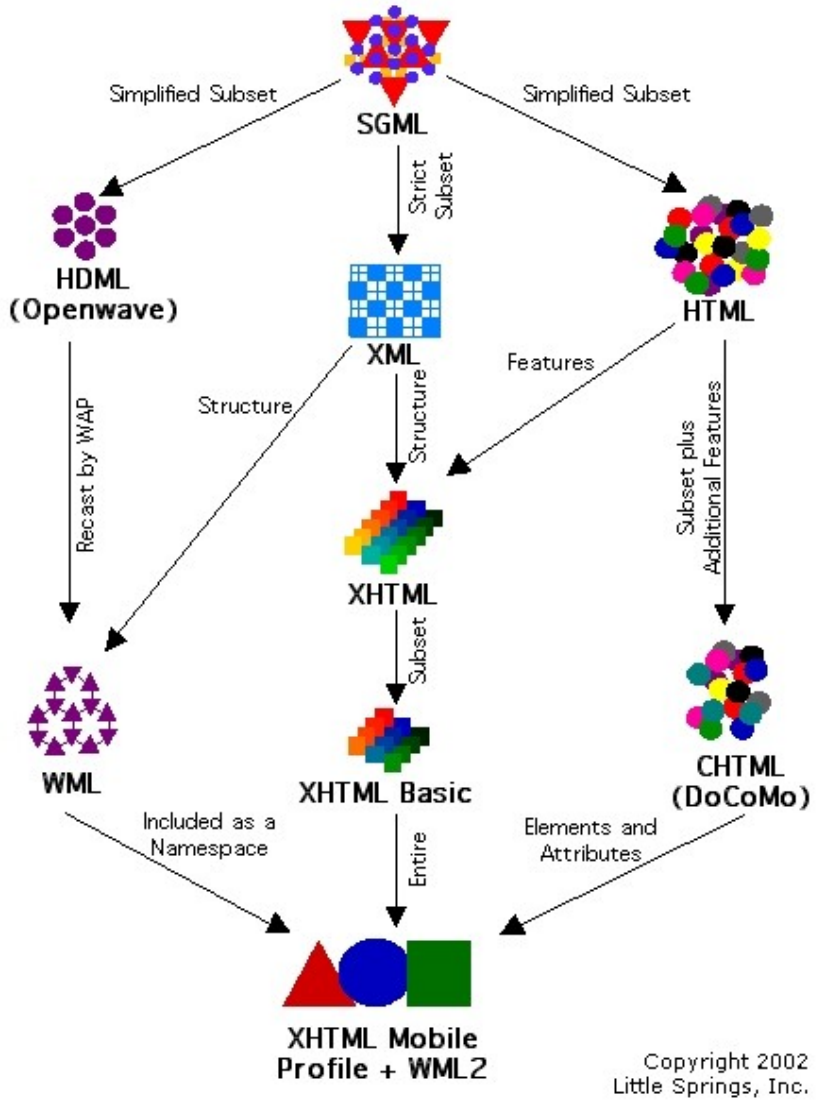
```
1  <?xml version="1.0"?>
2  <!DOCTYPE wml PUBLIC ".//WAPFORUM//DTD WML 1.1//EN"
3  "http://www.wapforum.org/DTD/wml_1.1.xml">
4  <wml>
5    <card id="Card1"  ontimer="#Card2" title="Prva stranica">
6      <p>Pozdrav od <b>WML-a!</b></p>
7        <timer value="50"/>
8    </card>
9    <card id="Card2" title="Druga stranica">
10     <p>
11         Nakon 5 sekundi, jos jedan pozdrav od WML-a!
12     </p>
13   </card>
14 </wml>
```


Deck-It WML Previewer 1.2.1
File
Options
My pages
Help
Web or WAP address:
http://161.53.67.199/report.xml
Source:
<?xml version="1.0" ?>
<RabbitData>
<Temperature>
28
</Temperature>
</RabbitData>
NOKIA
28
Back
Messages:
Generating wml...
Page not optimized for mobile
Parsing wml...
Displaying card
Page not optimized for mobile

Problemi vezani uz WAP 1.0

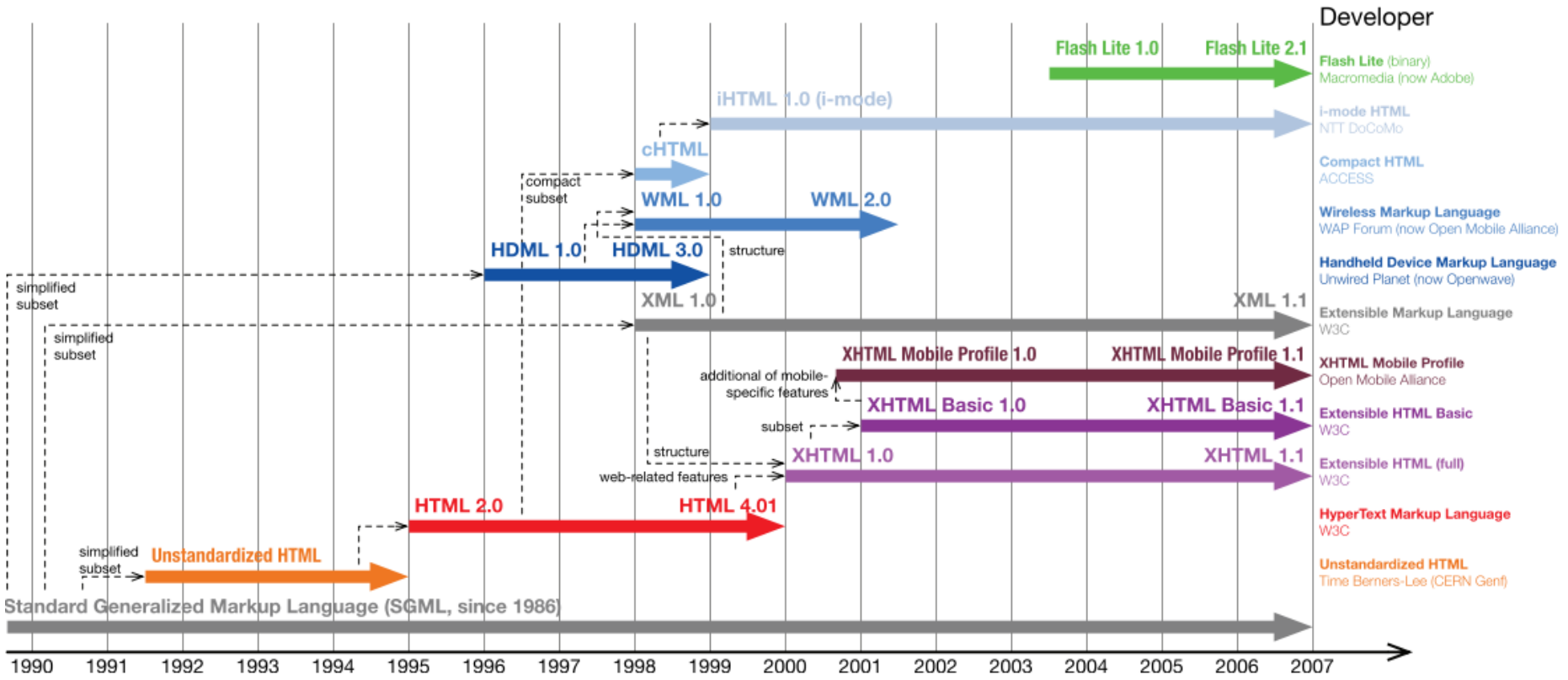
- Napravljen kao kombinacija više novih Internet protokola povezanih na nekompatibilan način
- Za prikaz sadržaja koristi se WML jezik koji nema veze sa označnim jezicima (HTML-om)
- Potreba za WAP gateway (nema transparentnosti između krajnjih korisnika)
- Napravio “paralelni svijet” - mobilni Web (LOŠE!)
- U biti, WAP 1.0 popularizirao je XML
- WAP 2.0 – poboljšanja, ...
  - jezici XHTML (Basic, MP, ...)

WML - XHTML


Simplified Subset
SGML
Simplified Subset
Strict Subset
HDML (Openwave)
XML
HTML
Features
Structure
Structure
Recast by WAP
Subset plus Additional Features
XHTML
Subset
WML
XHTML Basic
CHTML (DoCoMo)
Included as a Namespace
Entire
Elements and Attributes
XHTML Mobile Profile + WML2
Copyright 2002 Little Springs, Inc.

XHTML Mobile Profile

Evolucija "*Mobile Web-Related Markup Languages*"



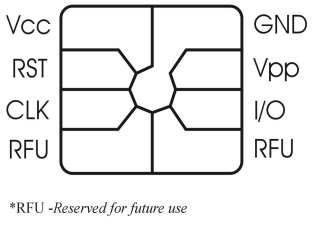
Java Card

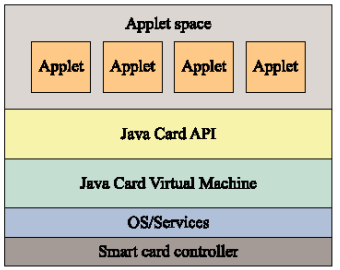
Java čip kartice (Java Card)

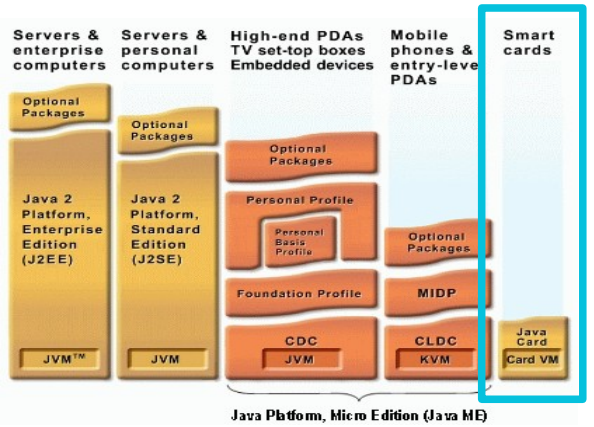
- kartice koje su sposobne pokretati Java applete
  - Specifikacija Java Card 2.0(http://www.javasoft.com/javacard)
  - API za Java kartice
- Appleti se pokreću u ograničenom sklopovlju
  - minimum 16K ROM-a, 8K EEPROM, 256 B RAM
- Java Card je podskup Jave

- Novo, novo,....
  - Java Card 3.0 specifikacija (Ožujak 2008.g.)
    - sve informacije za izgradnju Java Card virtualnog stroja (JCVM)

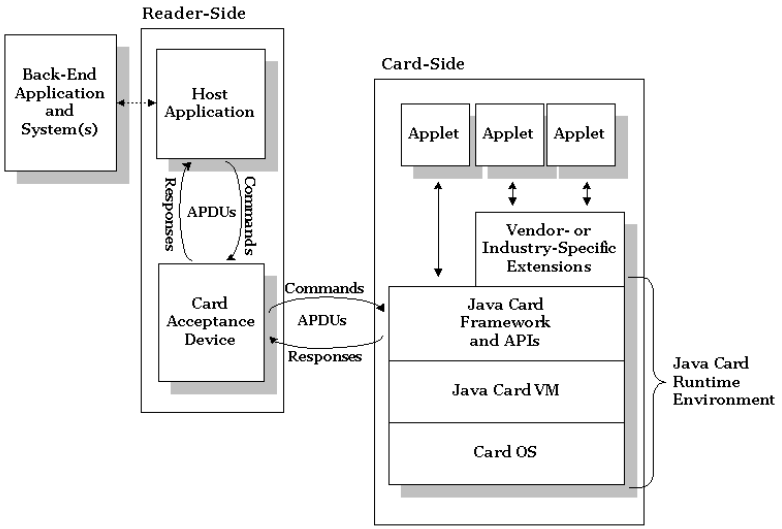
Java Card

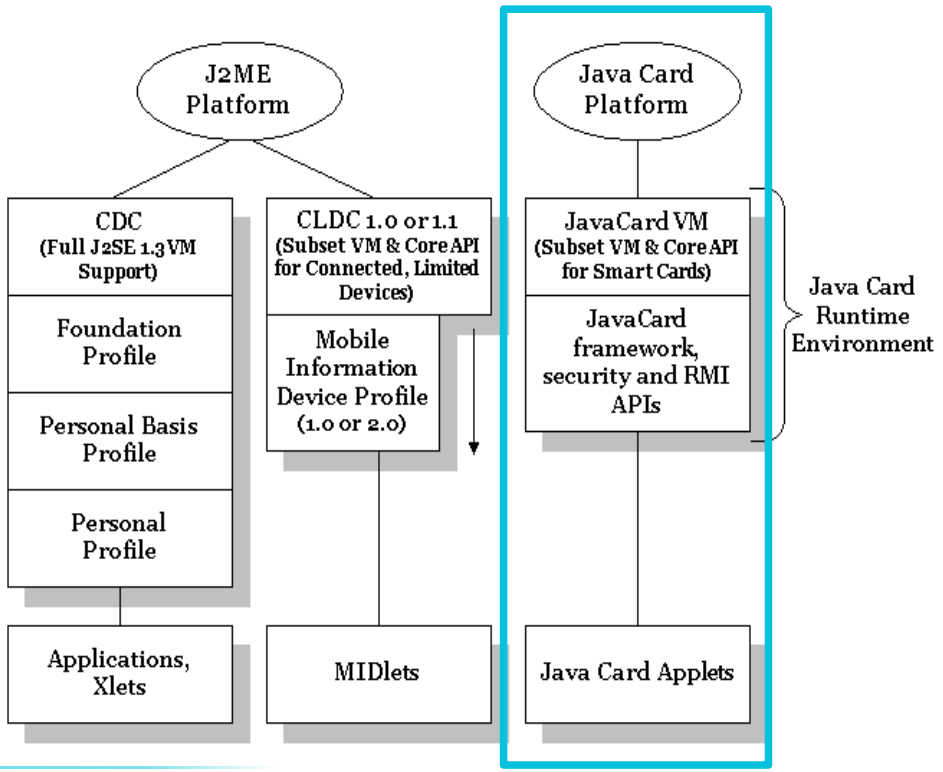

Vcc
RST
CLK
RFU
GND
Vpp
I/O
RFU
*RFU -Reserved for future use


Applet space
Applet
Applet
Applet
Applet
Java Card API
Java Card Virtual Machine
OS/Services
Smart card controller


Servers & enterprise computers
Servers & personal computers
High-end PDAs TV set-top boxes Embedded devices
Mobile phones & entry-leve PDAs
Smart cards
Optional Packages
Java 2 Platform, Enterprise Edition (J2EE)
Java 2 Platform, Standard Edition (J2SE)
Personal Profile
Personal Basis Profile
Foundation Profile
MIDP
CDC
CLDC
JVM™
JVM
KVM
Java Card
Card VM
Java Platform, Micro Edition (Java ME)


Die Karte


Reader-Side
Back-End Application and System(s)
Host Application
APDUs
Commands
Responses
Card Acceptance Device
Commands
APDUs
Responses
Card-Side
Applet
Applet
Applet
Vendor- or Industry-Specific Extensions
Java Card Framework and APIs
Java Card VM
Card OS
Java Card Runtime Environment


J2ME Platform
Java Card Platform
CDC (Full J2SE 1.3 VM Support)
Foundation Profile
Personal Basis Profile
Personal Profile
CLDC 1.0 or 1.1 (Subset VM & Core API for Connected, Limited Devices)
Mobile Information Device Profile (1.0 or 2.0)
JavaCard VM (Subset VM & Core API for Smart Cards)
JavaCard framework, security and RMI APIs
Java Card Runtime Environment
Applications, Xlets
MIDlets
Java Card Applets

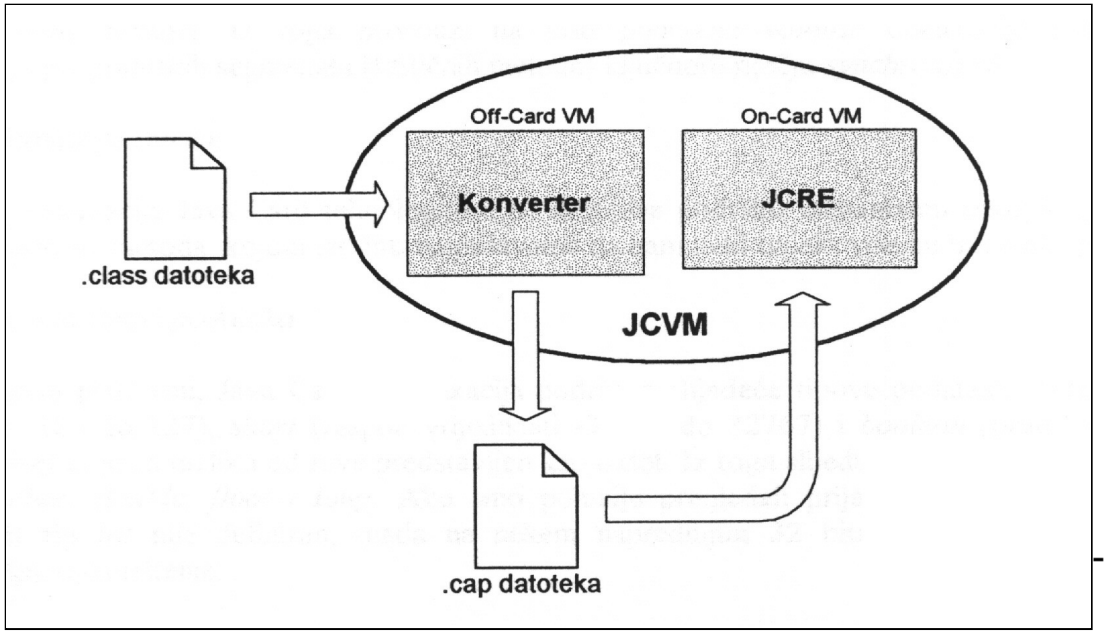
Java Card - životni ciklus

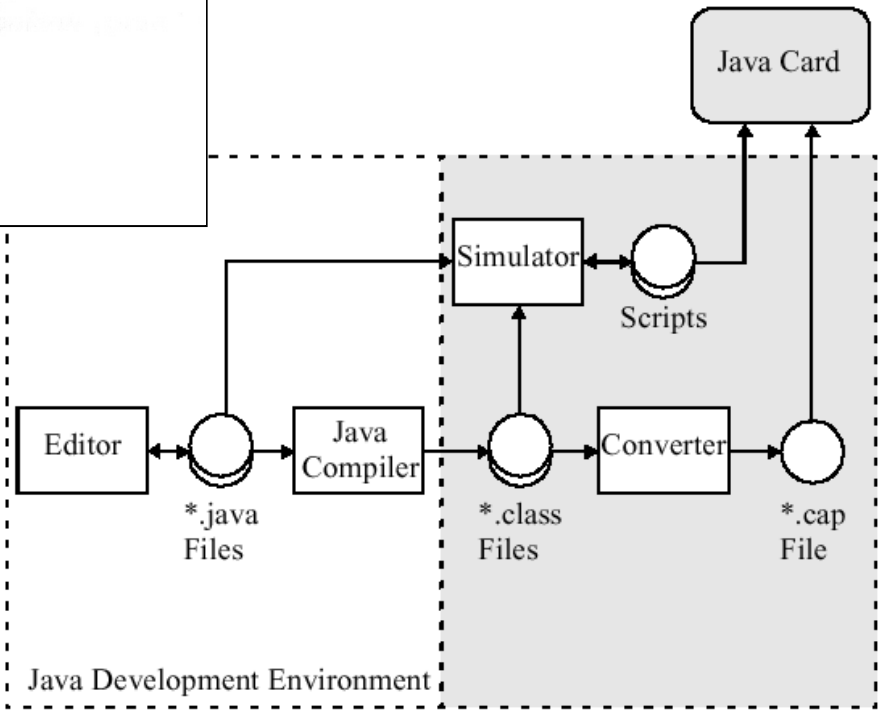
- 1. korak
  - životni ciklus počinje upisivanjem OS-a, JCVM-a, API-a i appleta u memoriju (ROM) - masking
- 2. korak
  - inicijalizacija (punjenje s podacima npr. podaci o izdavatelju (banka) i sl.
- 3. korak
  - personalizacija, dodjela kartice osobi (upis ključa, imena, šifre,.... u trajnu - EEPROM memoriju)
- Od tog trenutka nadalje (do uništenja, oštećenja, izlaska datuma valjanosti,...) kartica je aktivna

Java Card 2.0 framework

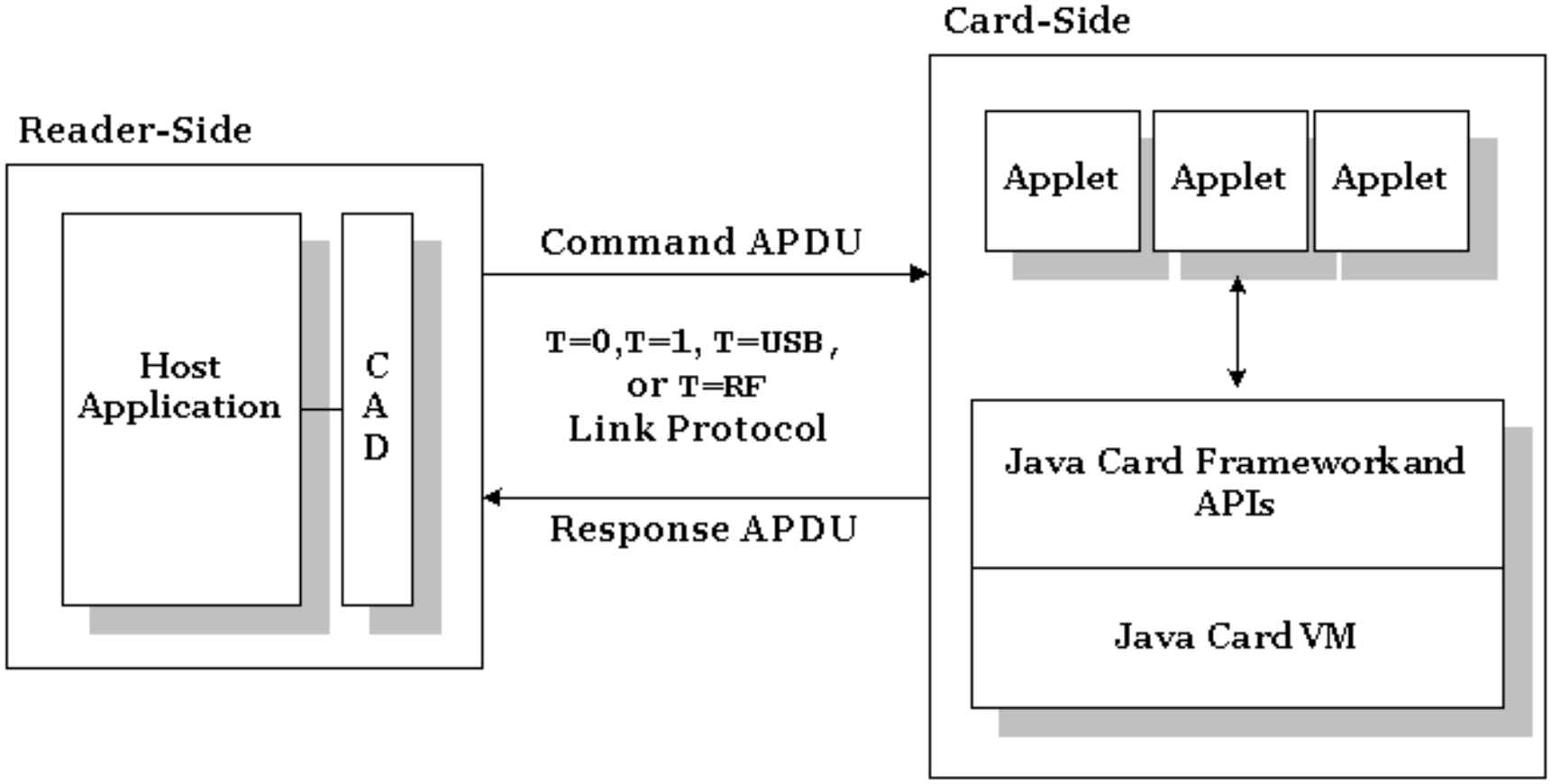
- Najčešće korištenje u bankarstvu
  - norme ISO 7816 i EMV - Europay, MasterCard, Visa
- Okvir (framework)
  - projektiran za olakšano korištenje i podršku aplikacijama
  - javacard.framework
    - osnovni razredi na kartici Applet, PIN
      - te APDU, System, Util - radna okolina
  - javacardx.framework
    - objektno orjentirani dizajn za ISO-7816-4 kompatibilan sustav datoteka
  - javacardx.crypto, javacardx.cryptoEnc
    - dva paketa za kriptografsku funkcionalnost kartice
  - paketi javacard*x* su proširenja okvira

Java Card - Razvojni ciklus za Applet





Komunikacijski protokol



http://developers.sun.com/mobility/javacard/articles/javacard1/fig-5.gif

Komunikacijski protokol

Command APDU						
Header (required)				Body (optional)		
CLA	INS	P1	P2	Lc	Data Field	Le

ISO 7816 CLA Values

CLA Value	Instruction Class
0x0n, 0x1n	ISO 7816-4 card instructions, such as for file access and security operations
20 to 0x7F	Reserved
0x8n or 0x9n	ISO/IEC 7816-4 format you can use for your application-specific instructions, interpreting 'X' according to the standard
0xAn	Application- or vendor-specific instructions
B0 to CF	ISO/IEC 7816-4 format you can use for application-specific instructions
D0 to FE	Application- or vendor-specific instructions
FF	Reserved for protocol type selection

Primjer koda

```
import javacard.framework.*;
...
public class CardTest extends Applet {

 //standard APDU input offset values
 public final static byte THIS_CLA = (byte)0x90;
 public final static byte INITIALIZE_TRANSACTION = (byte)0x20;
 public final static byte COMPLETE_TRANSACTION= (byte)0x22;
 public final static byte INITIALIZE_UPDATE= (byte)0x24;
 public final static byte COMPLETE_UPDATE= (byte)0x26;
....
private CardTest() {
    // ...
  register();
 }
....
public static void install(byte[] byteArray, short offset, byte length) {
  new CardTest();
 }
.....
```

Primjer koda - nastavak

```
/** Implementation of the standard method for processing an incoming APDU.*/
public void process(APDU apdu) {
 byte buffer[] = apdu.getBuffer();
 if (buffer[ISO7816.OFFSET_CLA] == THIS_CLA) {
  switch (buffer[ISO7816.OFFSET_INS])     {
   case INITIALIZE_TRANSACTION:
    writeBack(apdu, INIT_SEQUENCE);
    break;
   case COMPLETE_TRANSACTION:
    writeBack(apdu, COMPLETE_SEQUENCE);
    break;
   case INITIALIZE_UPDATE:
    writeBack(apdu, INIT_UPDATE_SEQUENCE);
    break;
   case COMPLETE_UPDATE:
    writeBack(apdu, COMPLETE_UPDATE_SEQUENCE);
    break;
   default:
    ISOException.throwIt(ISO7816.SW_INS_NOT_SUPPORTED);
  }
 }
}
```

Primjena Java Card tehnologije - korist

- Jednom napisani apleti mogu se koristiti na različitim karticama
- Na jednoj čip kartici može se pokretati više apleta
- Kad je već kartica izdana lako se nadograđuje apletima koji zadovoljavaju novonastale potrebe korisnika
- Objektno orijentirani ustroj Jave omogućuje fleksibilnost u programiranju
- Java Card API su sukladni s postojećim normama i specifikacijama

Prvi dio (off-card)

- Izvodi se na osobnom računalu
- Priprema učitavanje razreda i rješavanje referenci
- Dinamičko učitavanje razreda za vrijeme izvođenja nije podržano
  - limitirajući resursi
  - sigurnosni razlozi

Konvertor radi sljedeće:

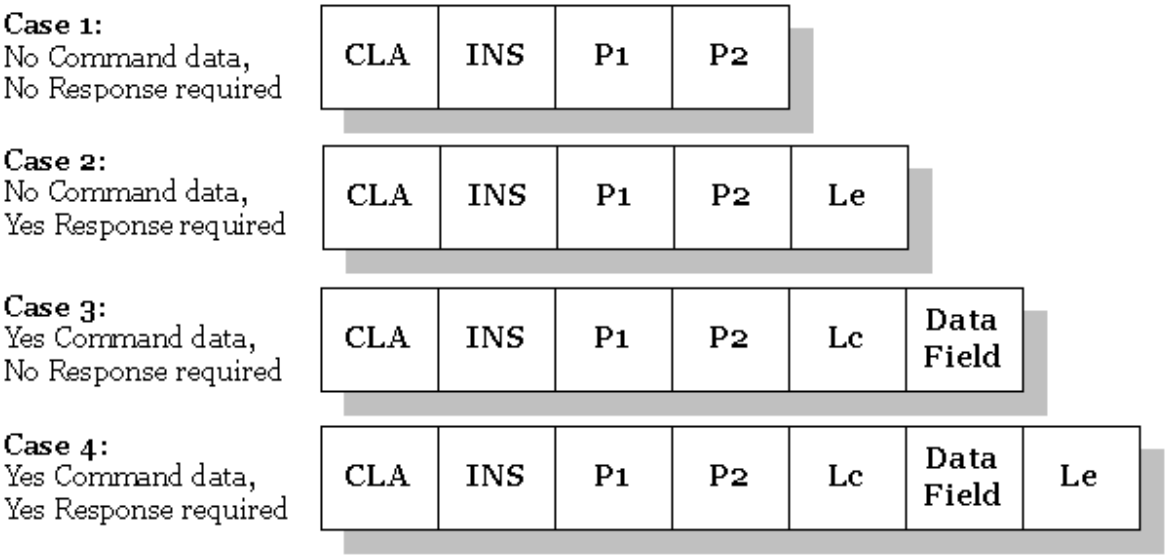
- Verifikacija:
  - da li su učitani razredi ispravnog formata s odgovarajućom tablicom simbola
  - da li je došlo do kršenja specifikacije jezika
- Priprema:
  - alocira i stvara podatkovne strukture za prikaz razreda, kreira statička polja i metode
  - inicijalizira statičke varijable na inicijalne vrijednosti
- Rezolucija:
  - određuje simboličke reference na razrede, metode i polja u kompaktnu formu kako bi se njima moglo bolje rukovati na čip kartici

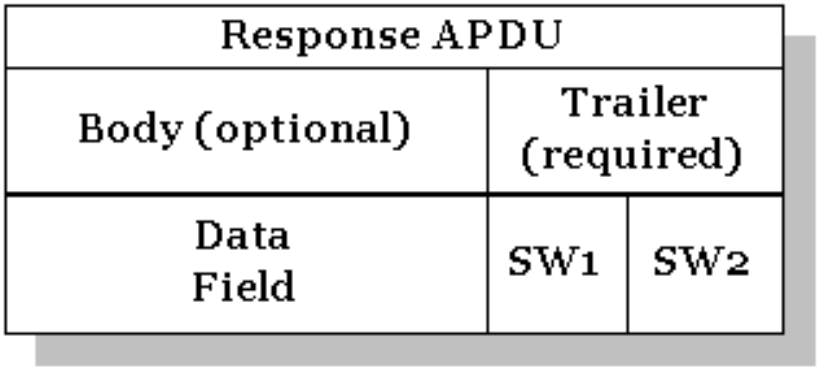
Drugi dio (on-card)

- Predstavlja bajt kod interpreter koji upravlja instaliranim razredima i objektima
- Provodi odjeljivanje apleta (firewall) i omogućuje sigurno dijeljenje zajedničkih resursa
- Životni vijek JCVM jednak je životnom vijeku čip kartice

Pitanja?

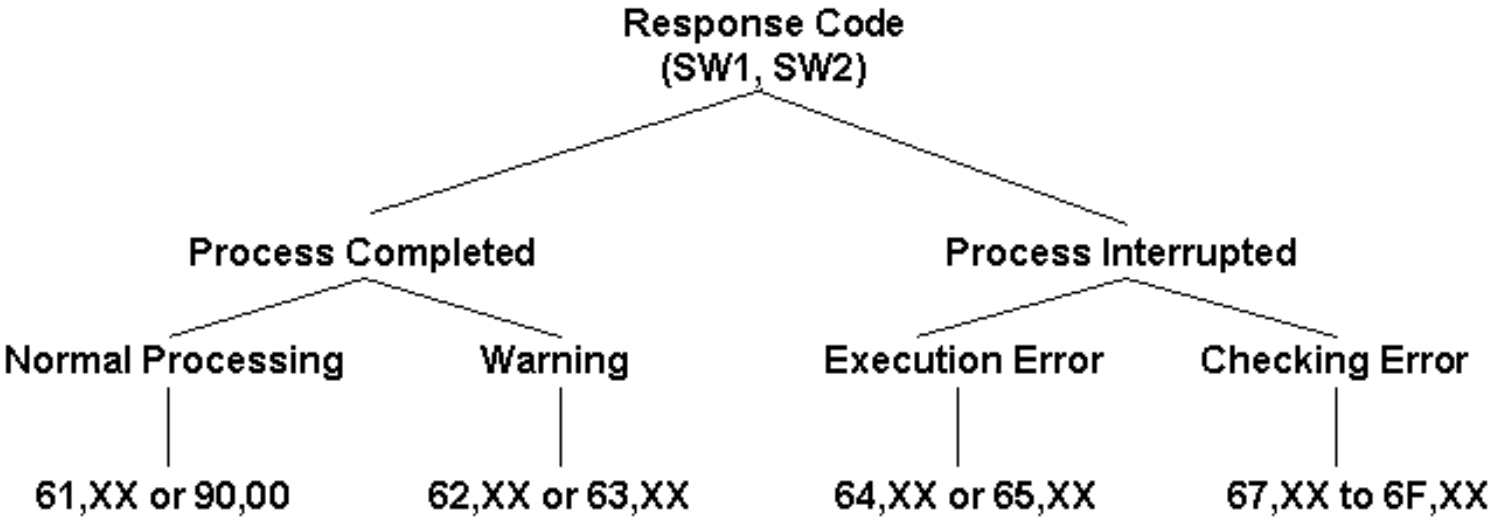
Komunikacijski protokol

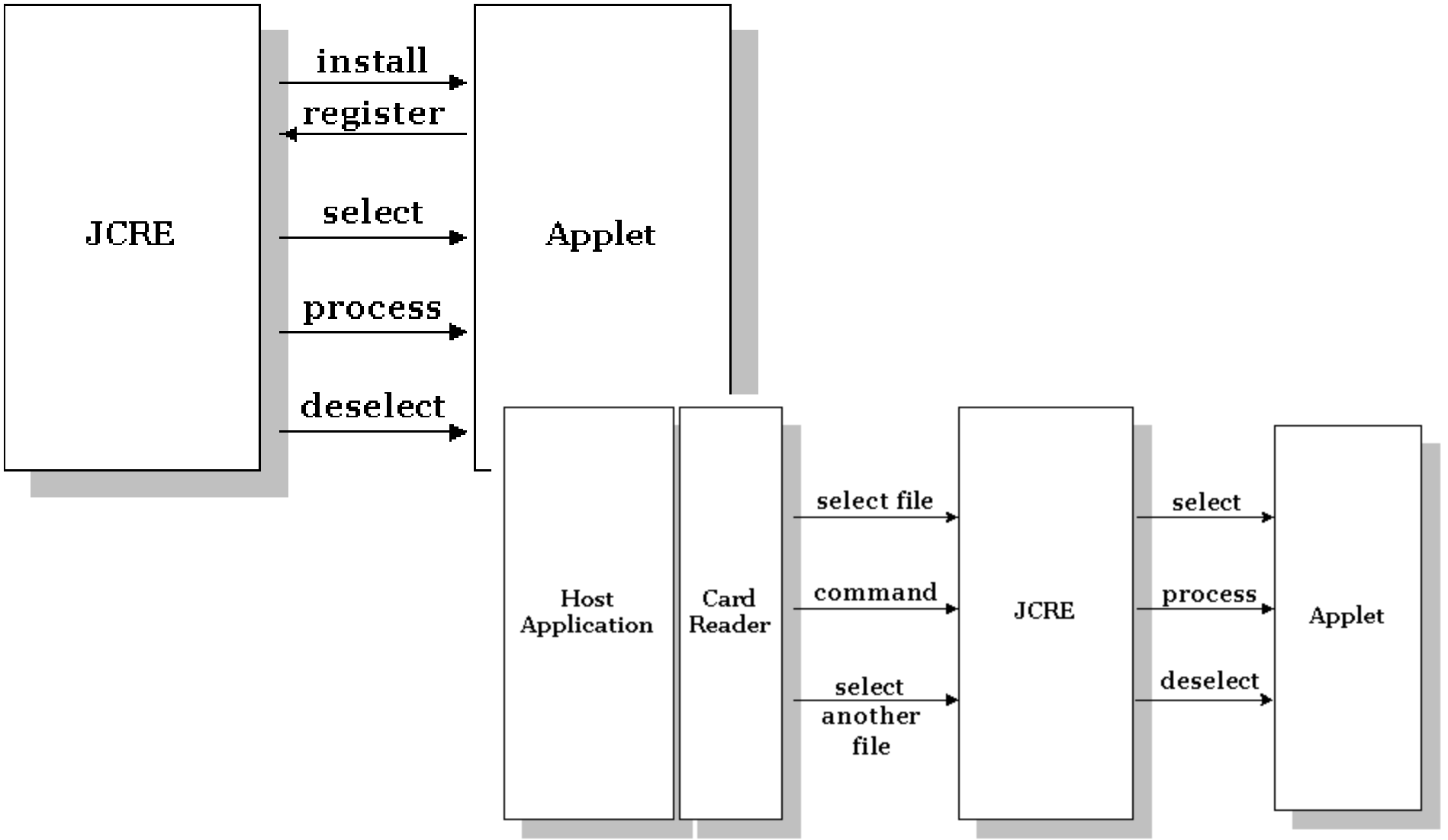


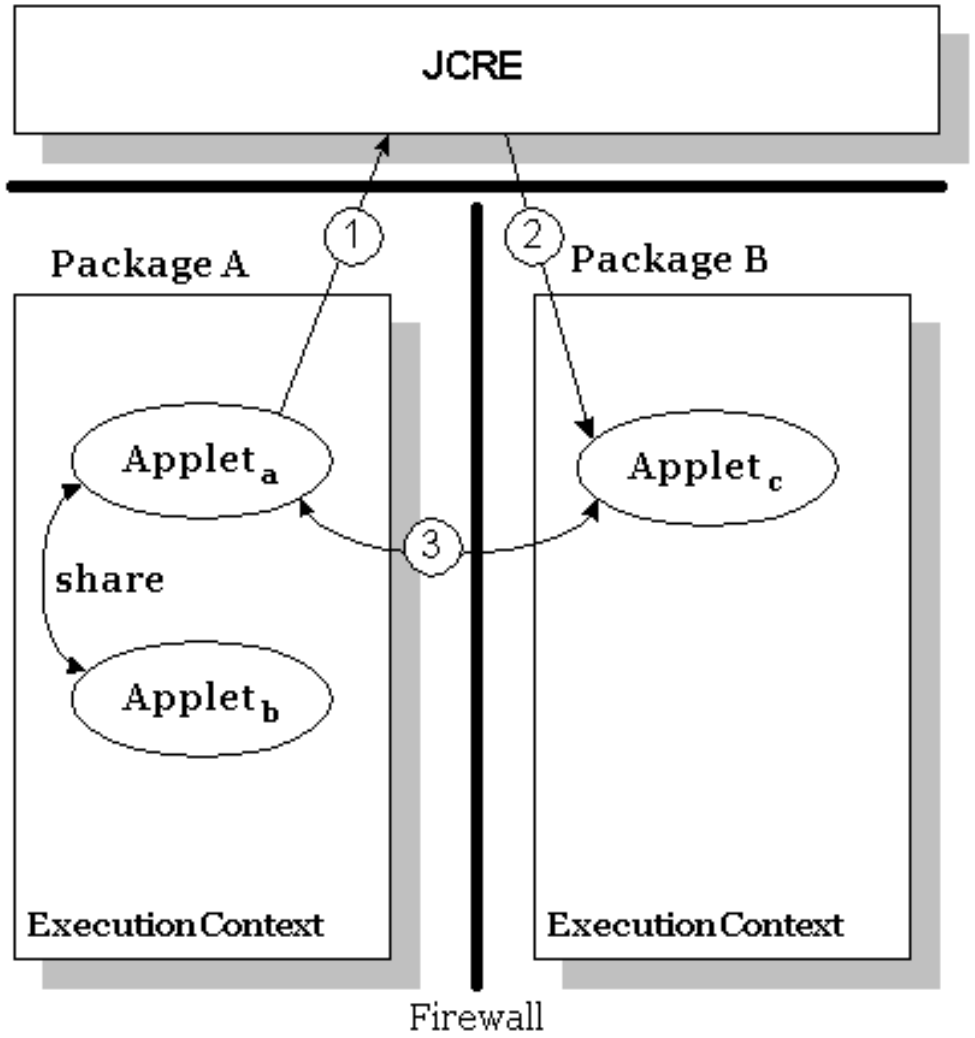


http://developers.sun.com/mobility/javacard/articles/javacard1/fig-7.gif

Komunikacijski protokol




JCRE
install
register
select
process
deselect
Applet
Host Application
Card Reader
select file
command
select another file
JCRE
select
process
deselect
Applet



An Introduction to Java Card Technology - Part 1
by C. Enrique Ortiz
May 29, 2003

Shema autentikacije