

RĪGAS TEHNISKĀ UNIVERSITĀTE  
Datorzinātnes un informācijas tehnoloģijas fakultāte  
Lietišķo datorsistēmu institūts  
Informātikas un programmēšanas katedra

**Pāvels KARTAŠEVS**

# **JAVA TEHNOLOĢIJAS MOBILO TELEFONU SPĒĻU IZSTRĀDĒ**

**Bakalaura darbs**

Darba vadītājs  
Mg.sc.ing, lektors  
Aloizs Ratnieks

Rīga, 2009



## Anotācija

Bakalaura darbā izpētīts, kā izmantojot Java ME tehnoloģiju var ērti izstrādāt spēles mobilām ierīcēm. Darbā parādītā mobilās spēles tēmas aktualitāte un pielietojumi.

Darbā tika dots priekšstats par Java ME tehnoloģijas komponentēm un to izmantošanu priekš mobilo spēļu izstrādes.

Pētījuma ietvaros tika salīdzinātas dažās integrētās izstrādes vides un tika izdarīti secinājumi par to piemērotības mobilo spēļu izstrādei. Salīdzinātas integrētas izstrādes vides NetBeans, EclipseME un rīku klāsts *Sun Wireless toolkit*.

Tika izpētīts mobilo spēļu tirgus. Saskaņā ar iegūtiem datiem, mobilo spēļu tirgus ļoti strauji attīstās. Mobilo ierīču lietotāji, ļoti bieži spēlē mobilās spēles un ielādē tos savās mobilās ierīcēs. Tika izpētīti mobilo spēles žanri. Saskaņā ar tiem gūts priekšstats par to, kādas spēles biežāk izmantotas.

Pētījuma ietvaros tika izveidota mobilā spēle ar piemērotāku integrēto izstrādes vidi.

Darba apjoms ir 58 lapaspuses, darbā iekļauti 26 attēli 7 tabulas, 3 pielikumi, 27 literatūras avoti.

## **Аннотация**

В работе бакалавра исследованно, как используя технологию Java ME можно легко разрабатывать игры для мобильных устройств. Показана актуальность данной темы и её применения.

В работе описана технология Java ME и её компоненты, и как их использовать для разработки мобильных игр.

В исследовательских целях сравнены среды разработки NetBeans и EclipseME, а также набор средств Sun Wireless toolkit для создания мобильных приложений.

В работе исследован рынок мобильных игр. Согласно найденным данным рынок развивается очень стремительно. Большинство пользователей загружают игры в свои мобильные устройства и используют их. Также описано большинство игровых жанров, дана информация о их популярности.

Как пример исследования была разработана игра с помощью выбранной, более подходящей среды разработки.

Работа содержит 58 страниц, 26 картинок, 7 таблиц, 3 приложения. Были использованы 27 литературных источников.

# Saturs

|   |    |
|---|----|
| IEVADS.....   | 6  |
| 1.JAVA ME TEHNOLOĢIJA .....   | 8  |
| 1.1 Java tehnoloģijas apraksts.....                                     | 8  |
| 1.2 JAVA ME platforma.....  | 9  |
| 1.3 Savienoto ierobežoto ierīču konfigurācija (CLDC).....               | 11 |
| 1.4 Mobilās informācijas Ierīču Profils.....                            | 13 |
| 1.5 Jaunumi MIDP 2.0 profilā.....                                       | 14 |
| 1.5.1 Pilnveidots Lietotāja interfeiss.....                             | 14 |
| 1.5.2 Multivides Atbalsts.....  | 14 |
| 1.5.3 Spēļu atbalsts .....  | 14 |
| 1.5.4 Paplašinātie savienojumi.....                                     | 14 |
| 1.5.5 Bezvadu savienojuma atbalsts.....                                 | 15 |
| 1.5.6 Galīga drošība.....   | 15 |
| 1.6 MIDlet.....   | 15 |
| 1.7 JAR faila formāts.....  | 16 |
| 1.8 MIDP 2.0 un 1.0 salīdzinājums.....                                  | 17 |
| 1.9 Net Beans, Eclipse izstrādes vide un Sun wireless toolkit rīki..... | 18 |
| 2.MOBILO SPĒĻU INDUSTRIJA.....  | 23 |
| 2.1 Mobilo spēļu žanri.....   | 23 |
| 2.1.1 Darbības spēles .....   | 23 |
| 2.1.2 Šaušanas spēles.....  | 23 |
| 2.1.3 Darbības-piedzīvojumu spēles.....                                 | 24 |
| 2.1.4 Simulācijas spēles.....   | 24 |
| 2.1.5 Citi spēļu žanri.....   | 25 |
| 2.2 Mobilo spēļu industrija.....  | 25 |
| 3.MOBILAS SPĒLES IZSTRĀDE.....  | 28 |
| 3.1 Mobilas spēles radīšana ar NetBeans izstrādes vidi.....             | 28 |
| 3.2 NetBeans vizuālie rīki spēles līmeņu izstrādei.....                 | 31 |
| 3.3 Java 2 ME klases lietošana spēles izstrādāšanā.....                 | 34 |
| 3.4 Izstrādātas spēles apraksts.....                                    | 37 |
| 3.4.1 Spēles izstrādes metodoloģijas apraksts.....                      | 38 |
| 3.4.2 Spēles struktūra.....   | 38 |
| 3.4.3 Spēles pirmkoda apraksts.....                                     | 38 |
| 3.5 Izstrādāta JAR faila struktūra.....                                 | 41 |
| 3.6 Lietotāja ceļvedis.....   | 42 |
| SECINĀJUMI.....   | 47 |
| LITERATŪRA .....  | 49 |
| PIELIKUMI.....  | 51 |

## IEVADS

Bakalaura darbs veltīts Java mobilo spēļu izstrādei. Tiks aplūkoti JAVA tehnoloģijas rīki ar kuriem tiks izstrādāta spēle, spēles izstrādes aspekti. Tiks aprakstīts J2ME profils, kas ļauj rādīt lietojumus mobilām ierīcēm. Darbā praktiskajā daļā tiks izstrādāta spēle priekš mobila telefona. Darbā iekļauta spēles izstrādes gaita.

JAVA tehnoloģija šodien ir ļoti populāra, jo tā ļauj rādīt platformu neatkarīgos lietojumus, tādēļ to izmanto lai izstrādātu lietojumus priekš mobilajām ierīcēm, mobilajiem telefoniem u.t.t. Mobilo telefonu platformas ir ļoti dažādās, tāpēc nepieciešams izmantot kādu universālu programmēšanas valodu, kas ļaus izstrādāt lietojumus neatkarīgi no platformas. Ražotāji ir atraduši visas šīs iespējas JAVA valodā un tika speciāli rādīts jauna programmēšanas valodas JAVA specifikācija J2ME speciāli priekš mobilajiem telefoniem.

Mobilo telefonu lietojumu klāstā ir ļoti populāras spēles. Šī popularitāte ir saistīta ar to, kā mobilais telefons ir vienmēr pa rokai un aktīvākie mobilo telefonu lietotāji - jaunie cilvēki aktīvi izmanto to izklaidei. Līdz ar to radusies veselā mobilo telefonu spēles industrija. Dažādas kompānijas nodarbojās ar spēļu izstrādi un pārdod spēles par naudu. Tāpēc mobilo spēļu izstrāde ievērojot ļoti plašu potenciālo pircēju skaitu ir pārvērtusies par labu biznesu dažādažām kompānijām.

Pārsvārā spēles priekš mobiliem telefoniem tiek klonētas no personālo datoru spēlēm. Izstrādātāji bieži pērk licences uz spēļu nosaukumiem un izstrādā līdzīgas spēles. Mobilo telefonu veikspēja ir daudz zemāka, nekā personālo datoru veikspēja, tāpēc mobilo telefonu spēles aptuveni atrodas līmenī kurā atradās spēles 90-x gadu sākumā personāliem datoriem.

Mana darba galvenais mērķis ir dot priekšstatu par Java ME tehnoloģiju un mobilās spēles izstrādi.

Darba uzdevumi:

1. Izpētīt JAVA ME tehnoloģiju, JAVA valodas J2ME profilu, kas apraksta lietojumu izstrādi mobilajām ierīcēm.
2. Veidot priekšstatu par mobilo telefonu spēlēm.
3. Izpētīt mobilo spēļu industriju.
4. Izveidot spēli priekš mobilā telefona.

Pirmajā nodaļā (Java ME tehnoloģija) ir dots Java ME tehnoloģijas un tas komponentu apraksts, salīdzinātas divas MIDP profilu versijas. Salīdzinātas integrētās izstrādes vides EclipseME un NetBeans.

Otrajā nodaļā (Mobilo spēļu industrija) apskatīti dažādi mobilo spēļu žanri, izpētīta

mobilo spēļu industrija, doti secinājumi par aktualitāti.

Trešajā nodaļā – Mobilas spēles izstrāde, dots izvēlētas integrētas izstrādes vides NetBeans rīku apraksts, kas ļauj ērti veidot mobilas spēles. Izveidots Java paketes apraksts, kas atbild par mobilas spēles radīšanu, apskatīta pētījuma ietvaros izveidota spēle.

3 Pielikumā ir aprakstīti sastopamākie termini.

# 1. JAVA ME TEHNOLOĢIJA

## 1.1 Java tehnoloģijas apraksts

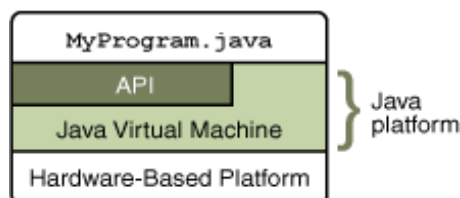
Java tehnoloģija iekļauj programmēšanas valodu un platformas. Java programmēšanas valoda tika radīta Sun Microsystems korporācijā, 1995 gadā. Java ir augsta līmeņa programmēšanas valoda – tā ir vienkārša, pārnesama, dinamiska, daudz-pavedienu un objektorientēta. Java programmēšanas valodas pirmkods tiek ierakstīts failos ar paplašinājumu .java. Pirmkods kompilējās ar Java kompilatoru .class failos, kuri satur platformas neatkarīgu baitu kodu. Baitu kods ir Java Virtuālas Mašīnas mašīnkods. Lai izpildītu kodu vispirms tiek palaista Java Virtuālas mašīnas instance, kas izpilda baitu kodu. Java virtuālā mašīna ir pieejama daudzām operētājsistēmām un datoru platformām.[SUN 1] Java lietojumiem ir ātra darbība pateicoties Java Virtuālās mašīnas iespējas veikt *JIT (Just In Time)* kompilāciju. Java virtuālā mašīna mijiedarbojas ar operētājsistēmu, piedāvājot pieeju pie failu resursiem un pie grafiskajiem resursiem.

Platforma sastāv no aparatūras vai programmu nodrošinājuma, kurā notiek programmas darbība. Vairākas platformas var tikt apskatītas, kā operētājsistēmas kombinācija ar aparatūru. (piemērām Windows platforma, Mac OS platforma) Java platforma atšķiras no tiem, tā ir tikai programmatūras platforma, kas tiek izpildīta uz aparatūras platformas virsmas. (sk. Attēlu 1)

Java platforma sastāv no divām komponentēm:

- Java Virtuālā Mašīna
- Java lietojumprogrammu saskarne

Java lietojumprogrammu saskarne ir liela programmatūras komponentu kolekcija, kas grupēta klašu un interfeisu bibliotēkas. Tādas bibliotēkas ir zināmas kā pakotnes. [SUN 1]



1. att. Lietojumprogrammas saskarne un  
Java Virtuālā mašīna atdala Java  
lietojumu no aparatūras



Sun Microsystems izstrādāja 3 platformas priekš dažādam darbības sfērām:

- Java 2 Enterprise Edition – platforma lai radītu servera lietojumus
- Java 2 Standart Edition – platforma personāliem datoriem
- Java 2 Micro Edition – platforma orientēta uz darbu ar mobilām ierīcēm

JAVA platformas mikro izdevums (*Micro edition*) piedāvā visas iespējas lai izstrādātu programmatūru mobilajām ierīcēm. Nekāda cita tehnoloģija nepiedāvā tik daudz lietojumprogrammu klāstu mobilajiem tālruņiem, kabatas datoriem un citām mobilām ierīcēm.[SUN 2]

## 1.2 JAVA ME platforma

Java ME platforma (*Java ME*) ir tehnoloģiju un specifikāciju kolekcija lai radītu platformu neatkarīgos lietojumus plašām mobilo ierīču skaitam – iebūvētām ierīcēm, mobilajiem tālruņiem, plaukstdatoriem un citām mobilajām ierīcēm.

Java ME tehnoloģija bija radīta lai izvairītos no ierobežojumiem, kas asociējās ar lietojumprogrammu izveidošanu priekš mazām ierīcēm. Ar Java ME tehnoloģiju var radīt Java lietojumus mazām ierīcēm ar ierobežotu atmiņu, displeju, jaudas un enerģijas kapacitāti. [SUN 3]

JAVA ME tehnoloģija ir balstīta uz 3 elementiem:

- Konfigurācija kas paredz vienkāršu bibliotēku klāstu un virtuālas mašīnas spējas plašam ierīču klāstam.
- Profila – komplektu lietojumprogrammas saskarni (API) , kurš atbalsta šaurāku ierīču skaitu
- Opcionālās paketes – tehnoloģijas specifiska lietojumprogrammas saskarne (API)

Lai veidotu lietojumus priekš JAVA 2 ME platformas pielieto nevis *Java Developer Kit*, bet speciālu rīku klāstu (*Sun Java Wireless Toolkit for CLDC*, *Sun Java Toolkit for CDC*, *NetBeans* izstrādes vide,kā arī citu ražotāju rīkus). Java 2 ME lietojumi strādā uz vienas no divām virtuālajām mašīnām: *KVM* priekš *CLDC* un *CVM* priekš *CDC* (*Connected device profile*). [МОНАХОВ 2008]

Kad *Java 2 Micro Edition* (J2ME) ielādētā ierīcē, tad uz tās var tikt palaisti visi lietojumi, kas atbalsta J2ME bibliotēkas. Piemēram e-pasta lietojums var tikt palaists uz Palm, Motorola telefona vai uz kādas citas ierīces, kas atbalsta *Java 2 Micro Edition* (J2ME).

*Java 2 Micro Edition* (J2ME) kļuva ļoti populāra pateicoties divām lietām:

- modularitāte un paplašinājums

- Java 2 Micro Edition (J2ME) sniedz virtuālo mašīnu diapazonu, kura ir optimizēta dažādiem procesoru tipiem un atmiņas apjomam, kas atrodams patentētāju un iebūvētajā ierīču tirgū.

Lai atbalstītu elastību tika piedāvāti konfigurācijas un profili.

Konfigurācija definē minimālu platformu priekš horizontālās ierīču klasēm ar līdzīgu atmiņas un veiktspējas iespējām.

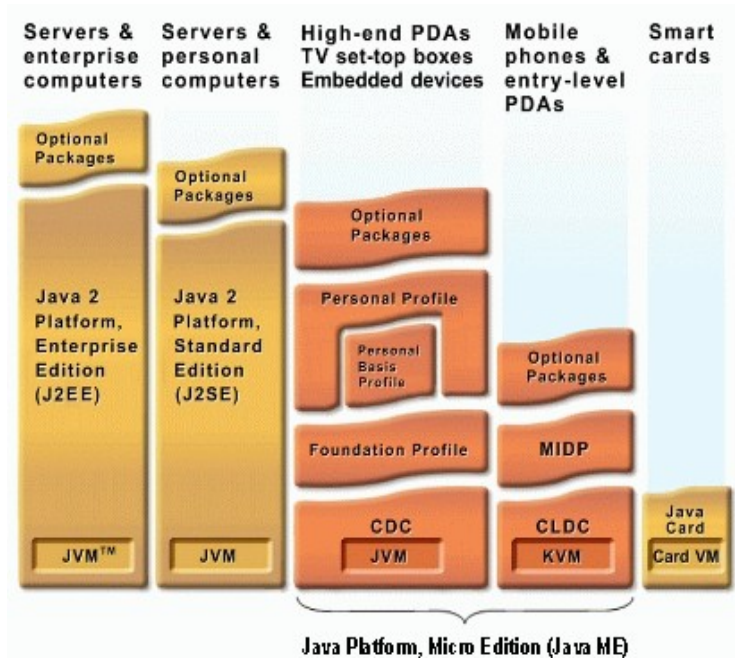
Profils definē Java platformu vertikālu tirgus segmentu. Katrs profils slāņots uz konfigurācijas virsotnes. Konfigurācija definē plašu ierīču klāstu, profils definē specializētu ierīču kategoriju ierīču klastā.

Lietojums kurš uzrakstīts specifiskajai konfigurācijai/profilam garantē pārnesamību uz visām ierīcēm kuras iekļautas konfigurācijas/profila asociācijā.

[Jafar Ajari 2001]

Ejot laikam „Java ME” platforma bija sadalīta divās konfigurācijās - viena mazām mobilām ierīcēm un viena mobilajām ierīcēm ar lielāku veiktspēju, tādām kā plaukstdatoriem un noteiktajām dekodētājierīcēm. Java Mikro izdevuma platforma tiek attēlota 2 attēlā.

Konfigurācija priekš mazajām ierīcēm tiek saukta par savienoto ierobežoto ierīču konfigurāciju (*Connected Limited Device Configuration*).

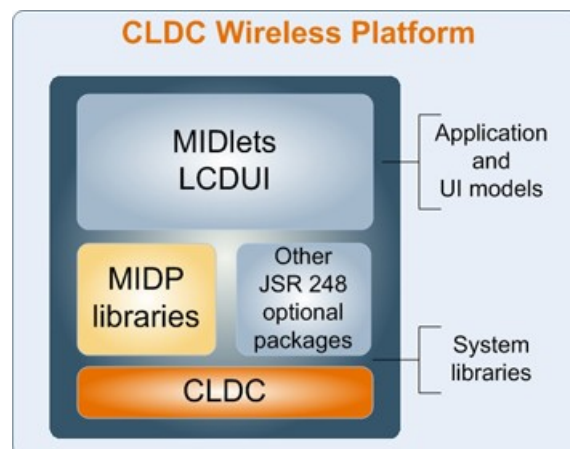


2 att. JAVA ME tehnoloģijas komponentes un to saistība ar JAVA tehnoloģiju [SUN 3]

### 1.3 Savienoto ierobežoto ierīču konfigurācija (CLDC)

Konfigurācija, kura izveidota ierīcēm, kurām ir ierobežoti resursi, kā mobilie telefoni, tiek saukta par savienoto ierīču konfigurāciju (turpmāk tekstā *CLDC*). Tā ir speciāli izveidota, lai palaistu Java platformu uz ierīcēm ar ierobežotām atmiņas, grafiskajām un veiktspējas resursiem. Java ME platforma arī specificē profilu klāstu, kuri noteic augstāka līmeņa lietojumprogrammas saskarni *API* un tālāko programmas definīciju. Praksē izmanto *CLDC* un *MIDP* kombināciju, lai nodrošināt pilnīgu Java lietojuma vides realizāciju mobilajiem telefoniem un citām ierīcēm ar līdzīgām iespējām. Lietojums tiek radīts, izmantojot dažādus lietojumprogrammas saskarnes (*API*) no profiliem. *CLDC* un *MIDP* kombinācijā tiek izveidots MIDlets. MIDlets ir lietojums(lietojuma struktūra), kurš tiek radīts ar Java ME tehnoloģiju(sk. 1.6 apakšnodaļu), tāds kā spēle, biznesa lietojums. Tādi MIDleti var tikt uzrakstīti vienu reizi un pēc tam izpildīti jebkurā ierīcē, kas atbalsta Java ME tehnoloģijas specifikāciju. MIDlets var atrasties arī kaut kur ārēja glabātuvē un lietotājs var lejuplādēt to izmantojot bezvadu pieslēgumu. [SUN 3]

CLDC platformas struktūra attēlotā 3 attēlā.



3.att. CLDC platforma

*CLDC* izmantojams uz ierīcēm ar nelielu veiktspēju. Ierīces parasti piemīt šāds raksturojums:[SUN 4]

- 16-bitu vai 32-bitu procesors ar takts frekvenci 16MHz vai augstāku
- No 160 KB atmiņas noteiktas priekš *CLDC* bibliotēkām un virtuālas mašīnas
- No 192 KB vispārējas atmiņas pieejamas Java platformai
- Zems enerģijas patēriņš, ierīce bieži tiek darbināta no baterijas
- Bezvadu savienojums

CLDC konfigurācija nodrošina bibliotēku bāzi un virtuālās mašīnas iespējas, kurām vajadzētu būt katrā J2ME vides implementācijā. Saistība ar vienu vai vairākiem profiliem CLDC veido spējīgu Java platformu, lai radītu lietojumus iebūvētām un pieprasītām elektroierīcēm.

Opcionalā pakotne ir tehnoloģiju specifiskas lietojumprogrammas saskarnes (API), kuras paplašina funkcionalitāti Java lietojumu vidēs. CLDC- bāzētas opcionālas pakotnes iekļauj bezvadu ziņojumu lietojumprogrammas saskarni (*Wireless Messaging API (WMA)*) un mobilās *Media* lietojumprogrammas saskarni (MMAPI).

CLDC specififikācijas mērķis – standartizēt augstas piemērotību, minimizēt apjomā Java lietojumu, izstrādes platformu resursu ierobežotām, bezvadu ierīcēm: [SUN 4]

- Samazināt lietojumu prasību līmeni priekš masveida tirga izvietojšanai
- Palielināt lietojuma pārnēsamību abstraktējot sistēmas operatorus standartizētai lietojumprogrammas saskarnei (*API*)
- Papildināt ierīču funkcionalitāti, atbalstot dinamisku lietojumu ielādi ierīcēs

Sistēmas resursu ierobežotības dēļ CLDC konfigurācijā netiek atbalstītas sekojošas īpašības, kas pieejamas JAVA 2 EE un JAVA 2 SE platformās:

- operācijas ar peldošo komatu (*floating point*)
- finalizācija (*finalization*)
- nav kļūdu atjaunošanas pēc sistēmas kļūdām (*error handling*)

Virtuālā mašīna, kura izmantojama CLDC konfigurācijā, nedaudz atšķīrās no virtuālās mašīnas JAVA, bet savienojama ar to. Virtuālā mašīna atrodas tieši mobilajā ierīcē un tiek saukta par kilobaita virtuālo mašīnu (*Kilobyte Virtual Machine*). KVM savas kompaktabilitātes dēļ ir daži trūkumi:

- Netiek atbalstītas operācijas ar peldošo punktu;
- nav klašu ielādētāja (*class loader*);
- nav atstarotāja mehānisma (*reflection*);
- nav realizēts *Java Nativ* saskarne;
- nav atbalstīta finalizācija;
- nav kļūdu atjaunošanas pēc sistēmas kļūdas (*error handling*);
- nav atbalstīts darbs ar grupas pavedieniem.

[Горнаков , 2004]

*The Connected Limited Device Configuration* (CLDC) ir standarta, pārnēsama, konfigurācija resursu ierobežotām ierīcēm un ir pamats mobilās informācijas ierīču profilam (*Mobile*

*Information Device Profile - MIDP).*

Profils definē lietojumprogrammas saskarnes (*API*) kopu ierīču klāstam un klašu bibliotēkas virs konfigurācijas. [Jafar Ajdari , 2001]

## **1.4 Mobilās informācijas Ierīču Profils**

Mobilās informācijas ierīču profils (MIDP) piedāvā standarta platformu mazām, resursu ierobežotām, bezvadu savienotām mobilajām ierīcēm.

Mobilās informācijas ierīču profils (MIDP) ir specifikācija, kura paredz galveno bibliotēku klāstu, lai rakstītu Java lietojumus mobilajām ierīcēm.

MIDP definē klases, lai radītu lietotāja saskarni, apstrādā lietotāja ievades datus, komunicēšanas caur tīklu ar HTTP protokolu.

Mobilās informācijas ierīču profils pievieno HTTP savienojuma interfeisu CLDC konfigurācijai. Tāds interfeiss satur metodes un konstantes, lai atbalstītu HTTP savienojumu, kurš var būt implementēts ar pārraides vadības protokolu/interneta tīkla protokolu (TCP/IP) vai ar bezvadu lietojumu protokolu (WAP) . [Jafar Adjari, 2001]

Tīkla programmēšanai var izmantot HTTP programmēšanas modeli un mobilie lietojumi strādās uz jebkuras mobilās ierīcēs, mobilā telefona ar WAP , plaukstdatora vai citas ierīces ar *Bluetooth*.

Šāds profils satur lietojumprogrammas saskarnes (*API*) klāstu un spēj radīt dažādus lietojumus no videospēlēm līdz pilnvērtīgām biznesa lietojumiem, kuri izmanto iekšējos un ārējos datu avotus.

Atzīmējami MIDP 1.0 ierobežojumi:

- nav aktīva renderēšanas lietojumprogrammu saskarnes (*API*) ;
- nav atbalsta priekš tiešas pikseļu zīmējuma piekļuves (RGB dati);
- nav pilnekrāna režīma atbalsta;
- nav audio atbalsta;
- Tikai HTTP atbalsts;
- Specifikācijas nav pilnīgi skaidras, var būt sarežģījumi to implementējot;
- Daži ierobežojumi var tikt novērsti, izmantojot ražotāja specifiskas lietojumprogrammas saskarnes (*API*) vai MIDP 2.0, bet tas samazina lietojuma pārnesamību [Wikipedia 1]

## 1.5 Jaunumi MIDP 2.0 profilā

### 1.5.1 Pilnveidots Lietotāja interfeiss

MIDP 2.0 uzlabo kopumā lietotāja pieredzi, pilnveido lietojuma pārnēsamību un piedāvā labāku paplašinājumu.

Jaunas funkcijas padarīs MIDP lietojumus vairāk interaktīvu, vieglāk izpildāmus.

Piemēri:

- Jauna “*Popup ChoiceGroup*” grupa rada tekošo izdalījumu kvadrāta ar vizuālu rindu;
- vienums (*Item*) ir to komandu klāsts un iespēja pievienot jaunas komandas;
- “*String Item*” un “*ImageItem*” var būt vairāk interaktīvi, pievienojot abstraktu komandu un noteikt izskata režīmu, lai pāradītu to kā hipersaiti vai pogu;
- Trauksmes ekrānam var būt abstraktas komandas, lai izstrādātāji varētu izmantot šo ekrānu, lai uzdotu lietotājam jautājumus.

MIDP 2.0 piedāvā vairāk lokāmu izkārtojumu (*layout*) vairākām lietojumu pārnēsamībām. Forma “*Screens*” piedāvā izstrādātājam vairāk kontroles pmobilajai ierīcei ekrāna izkārtojumā. Individuālas formas elementiem (vienumiem) ir minimālais un rekomendēts izmērs. Katra MIDP implementācija izmanto izkārtojuma un ekrāna atribūtus, lai optimāli pielāgotu lietojumu.

### 1.5.2 Multivides Atbalsts

MIDP 2.0 ietver Audio Celšanas Bloku (*Audio Building Block (ABB)*), kas arī ir daļa no *Mobile Media* lietojumprogrammas saskarnes (MMAPI), MIDP 2.0 opcionālas pakotnes. AAB dod iespēju izstrādātajai programmatūrai, pievienot audio, kā toņus, *wav* failus MIDP lietojumos bez MMAPi atbalsta. Ierīcēs, kuras atbalsta MMAPi implementāciju, izstrādātāji var pievienot vairāk multivides (*multimedia*) kontinenta.

### 1.5.3 Spēļu atbalsts

MIDP 2.0 pievieno spēļu lietojumprogrammas saskarni (API), kura piedāvā standarta pamatu, lai veidotu spēles. MIDP Spēļu lietojumprogrammas saskarne (API) ietver spēļu-specifisku funkcionalitāti, tādu kā spraiti, slāņi. Šāda funkcionalitāte atvieglo spēļu rādīšanu un pārvaldi.

### 1.5.4 Paplašinātie savienojumi

MIDO 2.0 pievieno servera ligzdu, seriāla porta, datagrammu komunikēšanas.

Paplašināta komunikēšana atvieglo integrāciju ar eksistējošo programmatūras infrastruktūru.

### 1.5.5 Bezvadu savienojuma atbalsts

Jaunā funkcija MIDP ar spēja dinamiski izveidot un atjaunot lietojumprogrammas caur gaisa bezvadu savienojumu (*over-the-air* (turpmāk *OTA*)). MIDP specifikācija piedāvā iespējas, ka jauni Midleti var būt instalēti, atjaunoti.

MIDP arī piedāvā iespēju servisa sniedzējām identificēt, kādi MIDletu komplekti strādā uz noteiktas ierīces un savākt atskaites no ierīces par instalēšanu, atjaunošanu vai dzēšanu.

MIDP OTA apgādes modelis nodrošina vienu standarta pieeju *MIDP* lietojuma izvietošanai, kurš strādā plaša mobilo ierīču diapazonā. *MIDP OTA* apgādes modelis ir definēts ierīču un servisu sniedzējiem, lai nodrošinātu uzticamu un drošu apgādes atrisinājumu.

### 1.5.6 Galīga drošība

MIDP 2.0 pievieno jauno drošības modeļi, balstītu uz atklātiem standartiem. Tas aizsargās tīklu, lietojumus un mobilās ierīces. MIDP 2.0 atbalsta HTTPS un SSL un WTLS standartus, lai pārraidītu šifrētus datus. MIDP 2.0 drošības domēni aizsargā no neautorizētas piekļūšanas datiem, lietojumiem un tīkla un ierīču resursiem. [SUN 44]

## 1.6 MIDleti

MIDlets ir JAVA lietojuma struktūra mobilās informācijas ierīces profilam (MIDP), kas parasti implementēta uz Java ME tehnoloģijas atbalstītas ierīces.

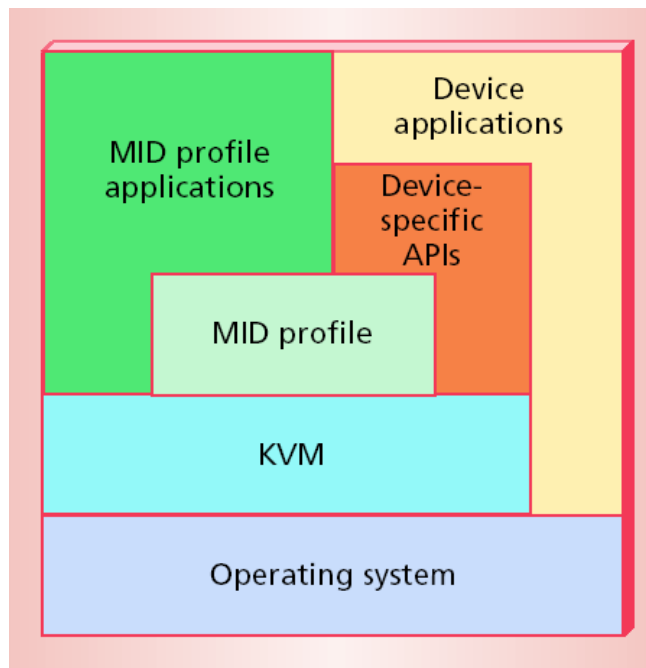
Midleta izplatīšanas galvenais fails ir .jar fails. *\_MIDleta* distributīvi var arī sastāvēt no .jad faila kas satur .jar faila satura izvietojumu. MIDleta implementācija var arī neprasīt .jad faila klātbūtni.

Lai izpildītos uz mobilā tālruņa Midletam jāatbilst šādiem prasījumiem:

- Galvenajai klasei jābūt apakšklasei *javax.microedition.midlet.MIDlet*
- MIDletam vajag būt sapaķotam iekšā .jar failā
- .jar failam vajag būt pre-verificētam izmantojot pirms verificētāju. (*preverification*)
- Dažos gadījumos .jar failam būt parakstītam mobilā telefona veidotajām

[Wikipedia 2]

Midlets ir līdzīgs appletam, tas satur lietotāja interfeisu, datus un kontroles iespējas. MIDlets ir JAVA lietojums, kas izstrādāts saskaņā ar CLDC un MIDP specifikācijām. 4. attēla parādītā Midleta izpildes atkarība no citām JAVA ME tehnoloģijas daļām.



4.att. KVM un MIDleta lietojumu sakarība

Appleta gadījumā zem programmatūra ir pārlūkprogramma. Midleta gadījumā programmatūra ir mobilais tālrunis, vai cita mobilā ierīce, kas atbalsta CLDC un MIDP.

MIDletu programmēt ir vienkāršāk salīdzinājumā ar *Java 2 Standard Edition*, tāpēc kā MIDP API ir vienkāršāks. [Jafar Ajdari, 2001]

Midleta izplatīšanas galvenais fails ir **.jar fails**. MIDleta distributīvi var arī sastāvēt no **.jad faila**, kas satur .jar faila satura izvietošanu MIDleta implementācija var arī neprasīt .jad faila klātbūtni.

## 1.7 JAR faila formāts

JAR tulkojams kā *Java Archive*. Tā faila formāts bāzēts uz populāra ZIP failu formāta un tiek izmantots, lai iekļautu vairākus failus vienā. JAR var tikt izmantots kā arhivēšanas rīks, bet galvenā tās izstrādes motivācija – lai Java lietojumi un to komponenti (.class faili, attēli, skaņas) varētu tikt lejuplādēti pārlūkprogrammā ar vienu HTTP transakciju. Tas paātrina ātrumu, ar kuru jaunais lietojums var tikt ielādēts un sākt funkcionēt. JAR formāts arī atļauj kompresijas iespējas, kas samazina faila izmēru un paātrina ielādi. Papildus JAR faila individuālie komponenti var tikt parakstīti ar tā autoru, lai nodrošinātu datu patiesumu.

JAR ir:

- Vienīgais kross-platformu failu formāts
- Vienīgais formāts, kas apstrādā audio, attēlu failus, kā arī .class failus



- Atpakaļsaderīgs ar eksistējošo appleta kodu
- Atvērtais standarts, viegli paplašinājams un uzrakstīts JAVA valodā
- Ērts veids, kā savienot java lietojuma daļas

JAR sastāv no PKWARE definēta zip arhīva, satur manifesta failu un potenciālus paraksta failus definētos Jar Faila specifikācija . [SUN7]

JAR faila manifests ir fails ar nosaukumu MANIFEST.MF. JAR faila manifests veidots no galvenās sekcijas, pēc kuras seko individuālas sekcijas, katra tiek atdalīta ar jauno rindiņu. Katrai sekcijai ir savi noteikumi. Galvenā sekcija apraksta JAR faila konfigurāciju, drošības informāciju, kā arī informāciju par papildus JAR failiem. [SUN 8]

JAR faili var arī saturēt *Classpath* ierakstu, kas identificē citus JAR failus lai tie ielādētos ar galveno JAR. Ieraksts sastāv no absolūtiem var relatīviem ceļiem, kas ved pie citiem JAR failiem.

Izstrādātāji var parakstīt JAR failus. Šajā gadījumā paraksta informācija ir iekļauta manifesta failā. Pats JAR fails nav parakstīts, bet katrs arhīva fails ir iekļauts ar savu kontrolsummu, kontrolsummas ir parakstītas. Daudzi var parakstīt JAR failu, izmainot JAR failu, bet parakstītie failu paliek derīgi. Kad Java lietojums izpildes laikā ielādē JAR failus, tās var validēt parakstus un atteikties ielādēt klases, kuras nesakrīt ar doto parakstu. Tādējādi var veidot “aizvērtus” JAR failus, kurās kļūdu ielādētās ielādēs tikai parakstītas klases. Tas novērsīs ļaunprātīga koda iekļaušanu eksistējošā pakotnē.[Wikipedia 4]

Izstrādātāji var arī paslēpt JAR failu informāciju, lai lietotājs nevarētu dabūt informāciju no koda, kā arī lai saīsinātu koda izmēru, kas noderīgs mobilos lietojumos.

## 1.8 MIDP 2.0 un 1.0 salīdzinājums

Kā ir redzams no tabulas 1, MIDP 2.0 ir MIDP 1.0 paplašinātā versija, bet tā arī resursu ietilpīgāka. MIDP 2.0 piedāvā plašu klāstu multivides (*multimedia*) iespēju, atvieglo programmēšanu, piedāvājot atbilstošus lietojumprogrammas saskarnes (API). piedāvā papildus drošības savienojuma protokolus.

| Minimālas prasības                                | MIDP 1.0   | MIDP 2.0  |
|---|--|---|
| Grafiskais displejs:                              | 96x54 pikseļi, ekrāna dziļums 1-bit,   | 96x54 pikseļi, ekrāna dziļums 1-bit,  |
| Ieejas mehānismi:                                 | Tastatūra, skārienjutīgs ekrāns  | Tastatūra, skārienjutīgs ekrāns   |
| Atmiņas resursi:                                  | 128 Kb energoneatkarīgas atmiņas priekš MIDP komponentēm<br>8 Kb energoneatkarīgas atmiņas lietojuma datiem,<br>32 Kb atmiņas Java virtuālajai mašīnai | 256 Kb energoneatkarīgas atmiņas priekš MIDP komponentēm<br>8 Kb energoneatkarīgas atmiņas lietojuma datiem,<br>128 Kb atmiņas Java virtuālajai mašīnai |
| Tīkla resursi:                                    | Divvirzienu, bezvadu savienojums   | Divvirzienu, bezvadu savienojums  |
| <i>Push</i> arhitektūra                           | Nav  | Ir  |
| Tīkla savienojumi                                 | <i>HTTP</i>  | + <i>HTTPS</i> , <i>SSL</i> atbalsts  |
| Multivides iespējas                               | Nav  | Wav faili, toņu atbalsts,   |
| Paplašinātais lietotāja interfeiss                | nav  | Ir  |
| Spēļu izstrādes lietojumprogrammas saskarne (API) | Nav  | Ir  |

MIDP 2.0 profils palīdz izveidot lietojumus ar lielāku funkcionalitāti, piemēram labākas spēles. Ar MIDP 2.0 ļoti viegli izstrādāt spēlēs, iekļauj sevī spēļu lietojumprogrammas saskarni (API).

## 1.9 Net Beans, Eclipse izstrādes vide un Sun wireless toolkit rīki

*Sun Java Wireless Toolkit* ir rīku klāsts lietojumu izstrādei, kuri balstīti uz J2ME savienoto ierobežoto ierīču konfigurāciju (*Connected Limited Device Configuration (CLDC)*) un mobilās informācijas ierīču profilu (*Mobile Information Device Profile (MIDP)*) . Ietver rīkus, kuri tiek izpildīti uz mobilajiem tālruņiem un citām mobilajām ierīcēm.

Rīku klāsts ietver sevī: [SUN 6]

- Dokumentāciju

- Lietojumu piemērus
- Mobilā telefona emulātorus
- Kompilātoru
- Optimizācijas un uzstādījumu rīkus

Sun Wireless Toolkit ir rīku un utilitātprogrammu kopums, kas ļauj veidot mobilo lietojumus. Lai veidotu mobilo lietojumus, izstrādātājam vajadzīgas labas programmēšanas prasmes, jo pirmkoda uzrakstīšana notiek teksta redaktorā.

NetBeans ir pilnvērtīga Java lietojumu izstrādes vide (skatīties 5 attēlu). Ar tās palīdzību var veidot profesionālus darbavirsmas, uzņēmuma, *web* un mobilo JAVA lietojumus.

Rādīt mobilo Java lietojumus var Mobilās informācijas ierīču profilam (*Mobile Information Device Profile (MIDP)*) 1.0, 2.0 versijai, Savienoto ierobežoto ierīču konfigurācijai (*Connected Limited Device Configuration (CLDC)* 1.0 un 1.1,) un Savienoto ierīču konfigurācijai (*the Connected Device Configuration (CDC)*).

Veidot grafisko lietotāja saskarni var ar vizuālo mobilo projektētāju: vilkt un nomest komponentes tādas kā gaidīšanas ekrāni, piekļuves ekrānus, faila pārlūkus, SMS rakstītājus, uzplaiksnījuma ekrāns. Ar analizētāja rīka palīdzību var samazināt faila izmēru, identificējot neizmantotos komponentus, pārbaudīt *MIDP* 1.0 savienojumu. Vizuālais mobila projekts arī ļauj vieglāk lokalizēt grafisko lietotāja interfeisu.

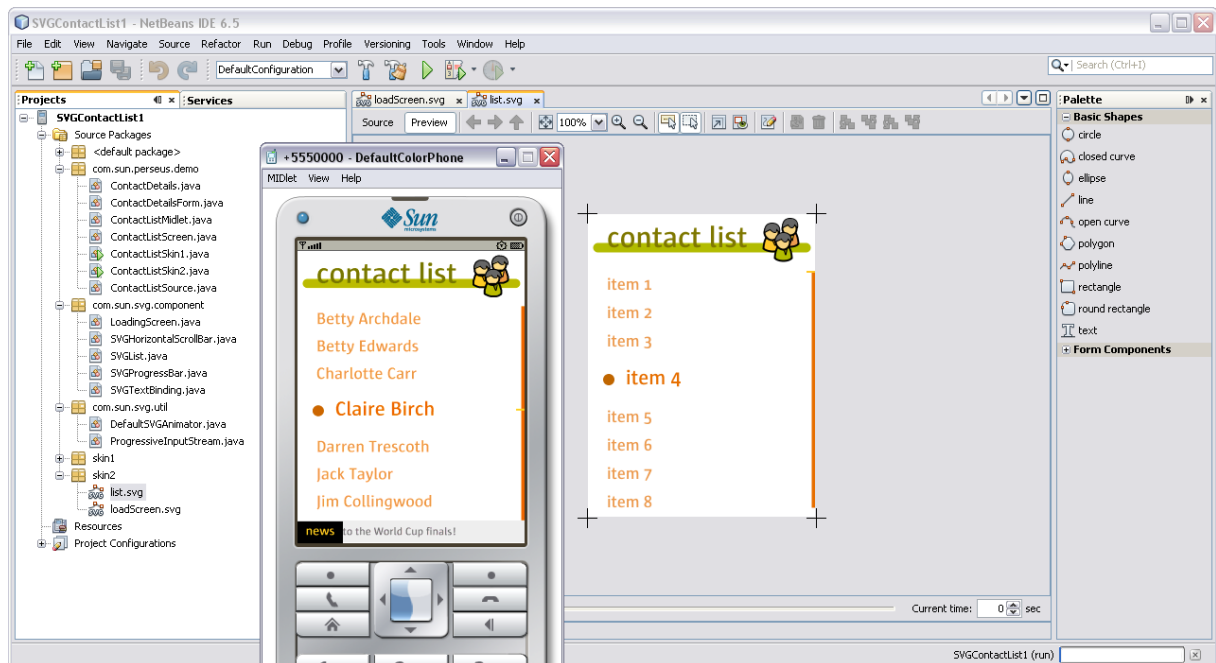
Datu saistīšana lietotāja interfeisa komponentēm ļauj lietojumiem veidot saikni ar displeju izmantojot datu saistīšanas vērtību redaktoru ar *Datu (DataSet)* komponentu no vizuālajā mobilā dizainera.

Veidot spēles ar vizuālo redaktoru, kas paredzēts *MIDP* 2.0 spēļu lietojumprogrammu saskarni (API). Lietojumprogrammu saskarne palīdz veidot spēles ar spraitiem uz spēles laukuma, izmantojot slāņu vadību.

Uzlabot izstrādes vidi pievienojot savus komponentus. Var pievienot jauno platformu tipus, konfigurācijas sniedzējus (*providers*).

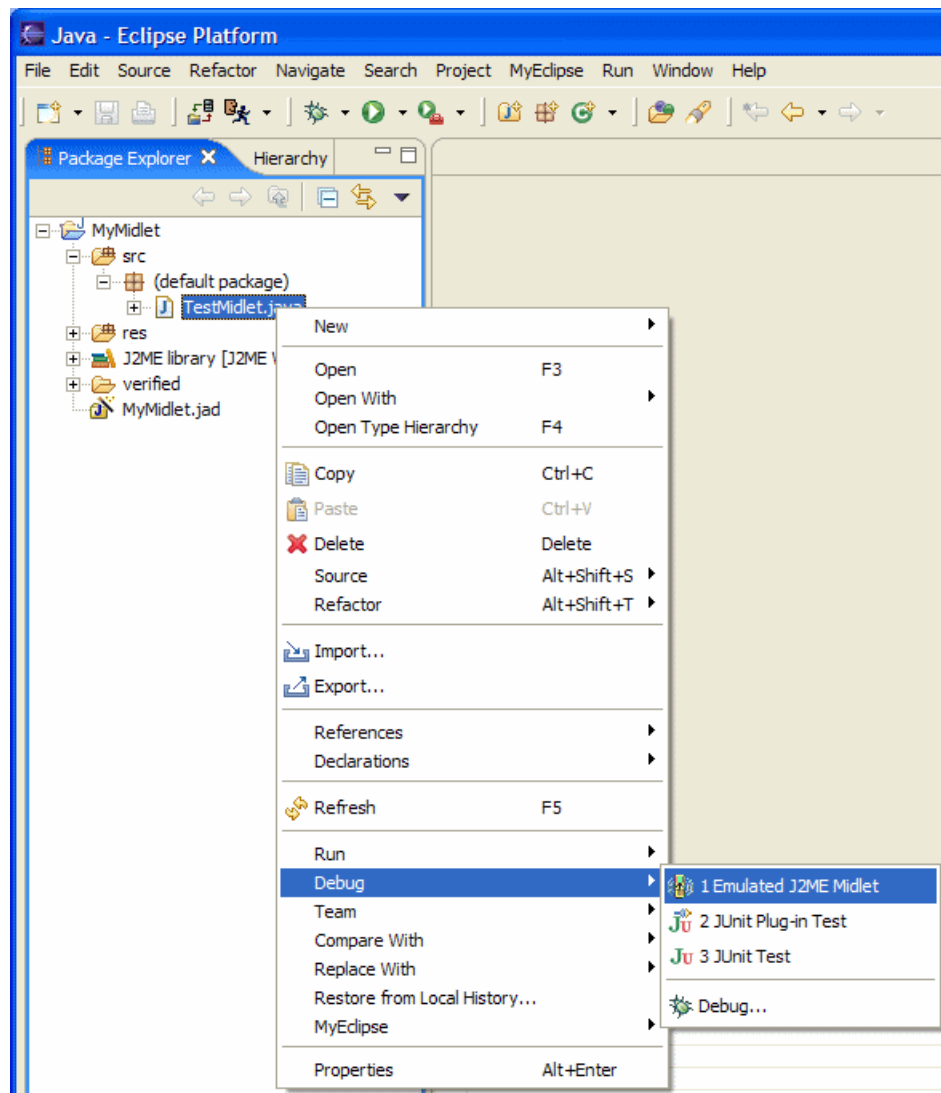
Integrētās izstrādes vides rīki lai testētu JUnit 1.1.0, Midletu parakstīšanu, sertifikātu valdību, automātisko koda noēnošanu ar *ProGuard* 4.2 rīku, integrētais gaisa savienojuma emulātors, “*push*” reģistra emulācija, WMA emulācija priekš SMS un CBS ziņojumiem, jaunie izvietošanas rīki, bezvadu ziņojumu un multimedija lietojumprogrammas saskarni (API). Izstrādātājs pilnīgi kontrolē šos rīkus ar *Apache Ant* skriptiem. [NETBEANS 1]

Kā redzams NetBeans ļoti labi der priekš mobila lietojuma izstrādes. Šajā integrētajā izstrādes vidē ir viss nepieciešamais, sākot no vizuālā izstrādes rīka (*Visual Mobile Designer*) attēlots 4 attēlā, līdz sintakses pārbaudīšanai un rīkiem, kas sastāda Midletu. NetBeans izstrādes videi nepieciešamais JDK distributīvs.



5.att. NetBeans vizuālā izstrādes vide

Līdzīga NetBeans videi, integrētā izstrādes vide Eclipse (parādīta 6 attēlā) arī piedāvā līdzīgas iespējas pateicoties “*Mobile Tools for Java*” un EclipseME spraudnim(*Plugin*). Pateicoties šiem komponentiem, Eclipse izstrādes vide var būt pilnīga, lai izstrādātu J2ME lietojumus, Midletus. EclipseME lietošanai vajadzīgi noteikti rīki kā “*Sun JAVA Wireless toolkit*” vai noteiktie mobilo ierīču ražotāju rīki.



6.att. Eclipse integrētā izstrādes vide [ECLIPSE 1]

EclipseME atbalsta:[DSDP Mobile Tools] [ECLIPSE 1]

- Midleta lietojuma projekta uzsākšanu
- JAR/JAD faila radīšana, izvietošana
- Midleta palaišana un atklādošana
- *Multi* konfigurāciju iespējas
- Midleta parakstīšana

Līdz ar to EclipseME atbalsta mazā funkciju priekš mobiliem lietojumiem, nekā NetBeans.

Izejot no visā aprakstošā seko, kā NetBeans ir vislielākā funkcionalitāte no visiem divām izstrādes vidēm, jo tai ir viena būtiska īpašība: Vizuālais mobila projekts atvieglo mobilo lietojumu veidošanu. Arī citi vizuālie izstrādes rīki palīdz izstrādāt spēles un spēles grafiskos elementus. Videi NetBeans ir kopīga struktūra ar EclipseME, kļūdu atklādošana,

koda rakstīšana, midleta apstrādes un veidošanas rīki. Dažādu izstrādes vides salīdzinājums tabulā 2.

2. tabula Sun Wireless Toolkit, EclipseME, NetBeans salīdzinājums

| Izstrādes vide, rīki<br>Vērtības | SUN Wireless Toolkit                    | EclipseME   | NetBeans                 |
|----------------------------------|---|---|--------------------------|
| Kompilators                      | ir                                      | Ir  | ir                       |
| Citi rīki priekš darbošanos      | nav                                     | <i>Mobile tools for Java</i> , kāds no rīkiem ( <i>Sun wireless toolkit</i> u.t.t.) | JDK                      |
| Programmas radīšana              | Nav rīku, var izmantot teksta redaktoru | Integrētā izstrādes vide  | Integrētā izstrādes vide |
| Ierīču emulātori                 | Ir                                      | Ir  | Ir                       |
| Jauno platformu radīšana         | nav                                     | Ir, paplašināma ar versijām   | Ir, konfigurējama        |
| Vizuālā midleta izstrādes vide   | nav                                     | nav   | ir                       |
| Projektu sastādīšana             | nav                                     | Ir  | Ir                       |
| Midleta izstrādes rīki           | ir                                      | ir  | ir                       |
| Projekta atklūdošana             | Iespējams tikai ar rokām                | Atklūdotājs iebūvēts  | Atklūdotājs iebūvēts     |

## 2. MOBILO SPĒĻU INDUSTRIJA

### 2.1 Mobilo spēļu žanri

Mobilo spēļu tipi ir tādi paši, kā personālo datoru spēles tipi. Viss, kas attiecas uz personālo datoru spēles tipiem, arī attiecas uz mobilo telefonu spēļu tipiem. (dažu mobilo spēļu piemēri parādīti pielikumā 1)

Spēļu žanru izklāsts:

#### 2.1.1 Darbības spēles

Darbības spēles (*Action*) prasa spēlētājam ātru reakciju un laiku, lai pārvarētu šķēršļus. Tas žanrs ir viens no pamata žanriem. Darbības spēlēm ir tendence uz cīņas elementiem. Darbības spēlēm ir daudzi apakšžanri, tādi, kā cīņas spēles, FPS.

Pirmā darbības spēle bija spēle Pong.

Darbības spēļu apakštipi:

- Novākt visus spēlētājus, koncentrēties uz daudzām tuvu cīņām, cīnoties ar daudzām datora kontrolētiem pretiniekiem vienlaikus. Spēle iekļauj spēlētāja cīņu ar dažādiem sarežģītības līmeņiem.
- Cīņas spēles (*Fighting games*) galvenais elements ir spēlētāju cīņa vienam pret vienu, viens var būt kontrolējams ar datoru.
- Labirinta spēles – spēles, kurām viss spēles lauks sastāv no labirinta, kurā spēlētāji izvairās no pretiniekiem, monstriem. Ātra domāšana un reakcijas laiks ieteicams taimera, monsturu, vai cita spēlētāja pienākšana finišam. Šādas spēles piemērs ir populāra spēle Pac-Man.
- “Pinball” spēles, spēlētājam tiek piešķirtas virtuālas “pinball” galdi, veidoti, lai atkartot galda spēles izskatu un sajūtu.
- Platformu spēles (*platformers*) Šādas spēles spēlētāji ceļo un lec starp dažādam platformām. Citi elementi paredz bēgšanu, kāpšanu pa kāpnēm. Platformeri bieži aizvieto elementus no citām spēlēm. (bieži cīņu, vai šaušanu)

#### 2.1.2 Šaušanas spēles

Šaušanas spēles fokusējas uz cīņu izmantojot ieročus, - pistoles, raketes. Tie dalās šaušanas spēlēs no pirmās sejas un trešās sejas. Atšķirībā no kameras perspektīvas.

1. Šaušanas spēles no pirmās sejas bieži saukti par FPS, piedāvā šaušanu un cīņu no

spēlētāja perspektīvas.

2. Sašaut visus ir šaušanas spēļu žanrs, kurā spēlētājs kontrolē personāžu vai mašīnu, bieži kosmosa kuģi un nošauj lielu ienaidnieku skaitu. Šāda žanra spēlēs prasa ātras reakcijas un ienaidnieku iegaumēšanu.
3. Taktiskās šaušanas spēles – variācijas ar šaušanas spēlēm fokusējas uz reālismu un uzsver taktisko spēlēšanu un komandas spēlēšanu.
4. Trešās personas šaušanas spēles uzsver cīņu un šaušanu no trešās personas perspektīvas, kad spēlētāja personāžs redzams no attāluma.

### **2.1.3 Darbības-piedzīvojumu spēles**

Darbības - piedzīvojumu (*Action/Adventure*) spēles kombinē šo divu žanru komponentes. Lielī šķēršļi kuri var tikt pārvarēti ar noteiktu rīku vai elementu, kurš tika savākts.

Pirmā darbības - piedzīvojumu spēle bija “Adventure” uz Altari 2600.

Piedzīvojumu spēles parasti tiek spēlētas virtuālajā pasaulē, kur loģiski savienotas istabas, ekrāni un objektīvi ir sarežģītāki, nekā vienkārši staigāt, lēkāt un ķert. Mērķi parasti tiek sasniegti pateicoties dažādu priekšmetu, atslēgu atrašanai, durvju atvēršanai. Personāžiem parasti ir inventārs: priekšmeti, atslēgas, rīki u.t.t.

[[ROBIN 1](#)]

### **2.1.4 Simulācijas spēles**

#### **1. Būvniecības un Biznesa(vadības) simulācijas spēles**

Šādas spēles ir simulācijas spēles, kurās spēlētāju uzdevums ir būvēt, paplašināt izdomātas projektus, ar ierobežotiem resursiem.

Biznesa simulācijas spēles ir mēģinājums stimulēt ekonomiku vai biznesu spēlētājam kontrolējot spēles ekonomiju

#### **2. Dzīves simulācija**

Spēlētājs kontrolē vienu vai vairākas mākslīgas dzīves, to attiecības. Var simulēt ekosistēmas, ko kontrolē spēlētājs.

#### **3. Transporta simulācijas spēles**

Spēles ar mērķi piedāvāt spēlētājam veikt reālistisku dažādu transportlīdzekļu vadīšanu. Piemērs: Sacīkstes, lidojumi, u.t.t.

4. Sporta spēles ļauj spēlētājam piedalīties kādā no sporta sacensībām, piemēram, futbolā.



### 2.1.5 Citi spēļu žanri

Stratēģijas (*Strategy*) fokusējas uz spēli kas, iekļauj domāšanu un plānošanu, lai panāktu uzvaru. Stratēģijas spēlētājs kontrolē savas vienības. Stratēģijas spēle var būt reāla laika vai soļu.

Galda spēles, karšu spēles piedāvā spēlēt attiecīgas spēles pret datoru.

Izglītības, loģiskas spēles – spēles, kuras paaugstina spēlētāja izglītības līmeni , piedāvājot tas atrisināt kādu mīklu (*Puzzle*), vai uzminēt vārdu.

[[PEABODY 1](#)], [[WIKIPEDIA 3](#)]

## 2.2 Mobilo spēļu industrija

Mobilas spēles kļuva ļoti populāras, pateicoties jaunajiem mobilajiem tālruņiem. Līdz ar to veidojās vesela mobilo spēļu industrija ar kompānijām, kuras ražo mobilās spēles un pārdod tās. Mobilo spēļu pārdošanas apjoms paaugstinās katru gadu.

Java mobilo spēļu tirgus aizņem aptuveni 3 miljardi dolāru, septiņi no 10 bezvadu lietojumiem uzrakstīti un lieto Java tehnoloģijas izpildes vidi.[[FINANCIAL 1](#)]

Pēdējo laiku statistika liecina par mobilo spēļu lieliem panākumiem tirgū: par ko liecina Telephia dati par Amerikas Savienoto Valstu 2005-2006 gadu statistiku.

Aptuveni 17.4 miljonu mobilo lietotāju lejuplādēja spēli 2006.gada 4.ceturksnī, par 45% vairāk nekā 2005 gadā, (sk. 3 tabulu) . Mobilo spēļu ieņēmumi no portālā pārdotām (Portāls - Tīmekļa vietne, kas piedāvā plašu resursu vai pakalpojumu klāstu ) ir 74% no kopējām spēļu ienākumiem, kamēr ne portālu ienākumi tikai 26%.

Mobilo spēļu tirgū pastāv liela konkurence ar 90 mobilo spēļu izdevējiem cīnoties par tirgus daļu. *EA Mobile* vada portāla spēļu lejupielādēs tirgu ar 28% ienākumu, *Gameloft* un *Glu Mobile* dala otro vietu ar 11 % katrā. *Namco*, *I-play*, *Digital Chocolate*, *Hands-On Mobile*, *Superscape*, *Capcom* un *Oasys Mobile* dala citās vietas ar ienākumu sadalījumu ar 2-6 procentu. [[NIELSENMOBILE 1](#)]

3. tabula lejuplādēs statistika un ienākumi ASV

|                           | Q4 2005 | Q1 2006 | Q2 2006 | Q3 2006 | Q4 2006 | % Pieaugums |
|---------------------------|---------|---------|---------|---------|---------|-------------|
| Lejupielādes skaits       | 12.0M   | 12.8M   | 13.5M   | 15.7M   | 17.4M   | 45%         |
| Pilnie ceturkšņa ienākumi | \$94M   | \$134M  | \$141M  | \$140M  | \$151M  | 61%         |

Zinotājs: Telephia mobilo spēļu atskaite , Q4 2006

Aplūkojot mobilo spēļu žanru datus par 2006.gada 2.ceturksni Lielbritānijā, var spriest par noteikto spēļu žanru popularitāti:

Mīklu/Stratēģijas (*Puzzle/Strategy*) un Arkādes (*Retro/Arcade*) mobilās spēles ir pašas

populārākās Lielbritānijā. Saskaņā ar Telephia's otrā 2006. gada ceturkšņa 3G Lielbritānijas abonentu atskaiti rāda, ka 61 procents mobilo spēļu spēlētāju lejupielādējuši Mīklu/Stratēģijas (*Puzzle/Strategy*) mobilo spēli, kamēr Arkādes (*Retro/Arcade*) spēles lejupielādēja tikai 45 procenti spēlētāju (skatīties tabulu 4). Darbībās/Piedzīvojumu (*Action/Adventure*) un Kāršu/Kazino (*Card/Casino*) spēles ieguvušas 42 un 39 procentus, kam seko Vārdu (*Trivia/Word*) (32%) un Sporta/Sacīkšu (*Sports/Racing*) (30%) spēles. Pēc Lielbritānijas mobilo sakaru operatora Vodafone 3G Mīklu/Stratēģijas (*Puzzle/Strategy*) mobilo spēles lejupielādējuši 70% spēlētāju. Vairāk nekā trešdaļa 3G abonentu ziņoja, ka tie ir lejupielādējuši spēles uz savas mobilo ierīces.

4. Tabula. Lielbritānijas abonentu 3G mobilo spēļu žanri, [Telephia U.K. 3G Subscriber Report, Q2 2006]

| Spēles žanrs   | Izplatības radītājs |
|--|---------------------|
| 1. Mīklu/Stratēģijas ( <i>Puzzle/Strategy</i> )      | 61%                 |
| 2. Arkādes ( <i>Retro/Arcade</i> )                   | 45%                 |
| 3. Darbībās/Piedzīvojumu ( <i>Action/Adventure</i> ) | 42%                 |
| 4. Kāršu/Kazino ( <i>Card/Casino</i> )               | 39%                 |
| 5. Vārdu ( <i>Trivia/Word</i> )                      | 32%                 |
| 6. Sporta/Sacīkšu ( <i>Sports/Racing</i> )           | 30%                 |

Lejupielādētas spēles lietotāji vairāk dod priekšroku nesaistes spēlēm, nekā tiešsaistes. Balstoties uz Telephia datiem, 7 no 10 abonentiem kuri spēlē mobilo spēles dod priekšroku lejupielādēt spēli uz viņu mobilo ierīci un spēlēt to nesaistes režīmā. (skatīties tabulu 5). Neskatoties uz optimizēto 3G savienojumu tiešsaistes spēles aizņem 5-7 %. [TELEPHIA 1]

5. tabula. Spēles veidu popularitāte starp Lielbritānijas 3G abonentiem

| Spēles tips                                 | Izplatības radītājs |
|---|---------------------|
| 1. Nesaistes                                | 71%                 |
| 2. Tiešsaistes, daudzlietotāju, gājienu     | 7%                  |
| 3. Tiešsaistes, daudzlietotāju, reāla laika | 7%                  |
| 4. Tiešsaistes, viena spēlētāja             | 5%                  |
| 5. Atrašanas vietas balstītas spēles        | 3%                  |

Jaunākie ASV dati arī liecina par to, ka redzams spēļu kategorijas tikai nedaudz atšķirās no Lielbritānijas spēļu žanra sadalījuma.

2008 gada 1 ceturksnis:

5 spēļu kategorijas no ienākumu puses. [TELEPHIA 2]

1. Mīklu/Stratēģijas (*Puzzle/Strategy*)
2. Darbības/Piedzīvojumu (*Action/Adventure*)
3. Sporta/Sacīkšu (*Sports/Racing*)
4. Kāršu/Kazino (*Card/Casino*)
5. Vārdu (*Trivia/Word*)

Dažādas mobilās spēles ir ļoti populāras, tās ielādē savā mobilajā ierīcē ļoti daudz cilvēku.

Dažu spēļu ienākuma daļa parādīta 6 tabulā. [TELEAPHIA 3]

6.tabula Mobilās spēles pēc ienākuma daļas ASV- Q3 2008 [TELEAPHIA 3]

| Spēles rangs | Spēle                              | Ienākuma daļa |
|--------------|------------------------------------|---------------|
| 1            | Tetris                             | 7.0%          |
| 2            | Bejeweled                          | 4.0%          |
| 3            | Guitar Hero III                    | 3.6%          |
| 4            | Wheel of Fortune                   | 2.6%          |
| 5            | PAC-MAN                            | 2.5%          |
| 6            | The Oregon Trail                   | 1.9%          |
| 7            | Ms. PAC-MAN                        | 1.7%          |
| 8            | Are You Smarter Than A 5th Grader? | 1.6%          |
| 9            | Tetris Mania                       | 1.6%          |
| 10           | Surviving High School              | 1.2%          |

Pēc visiem šīm datiem var spriest, kā mobilās spēles ir ļoti populāras, tās ielādē ļoti daudz cilvēku dažādās vecuma kategorijās. Mobilo spēļu izstrādātāji un izplatītāji pelna ļoti daudz naudas uz mobilajām spēlēm. Katru gadu mobilo spēļu tirgus aug un izplatās. Tas mudina izstrādāt jaunas un sarežģītākas spēles vai spēles ar jaunām grafiskām iespējām. Arī jaunas jaudīgākas mobilas ierīces palīdz vēl vairāk izplatīt mobilas spēles. Tiek veidoti jaunie rīki ar kuriem var vieglāk un ar mazākiem patēriņiem izstrādāt jaunas spēles, mobilo spēļu reklamē izplatītāji, kā arī mobilo sakaru sniedzēji, lai darītu zināmu plašai auditorijai par jauno spēļu iznākšanu.

### 3. MOBILAS SPĒLES IZSTRĀDE

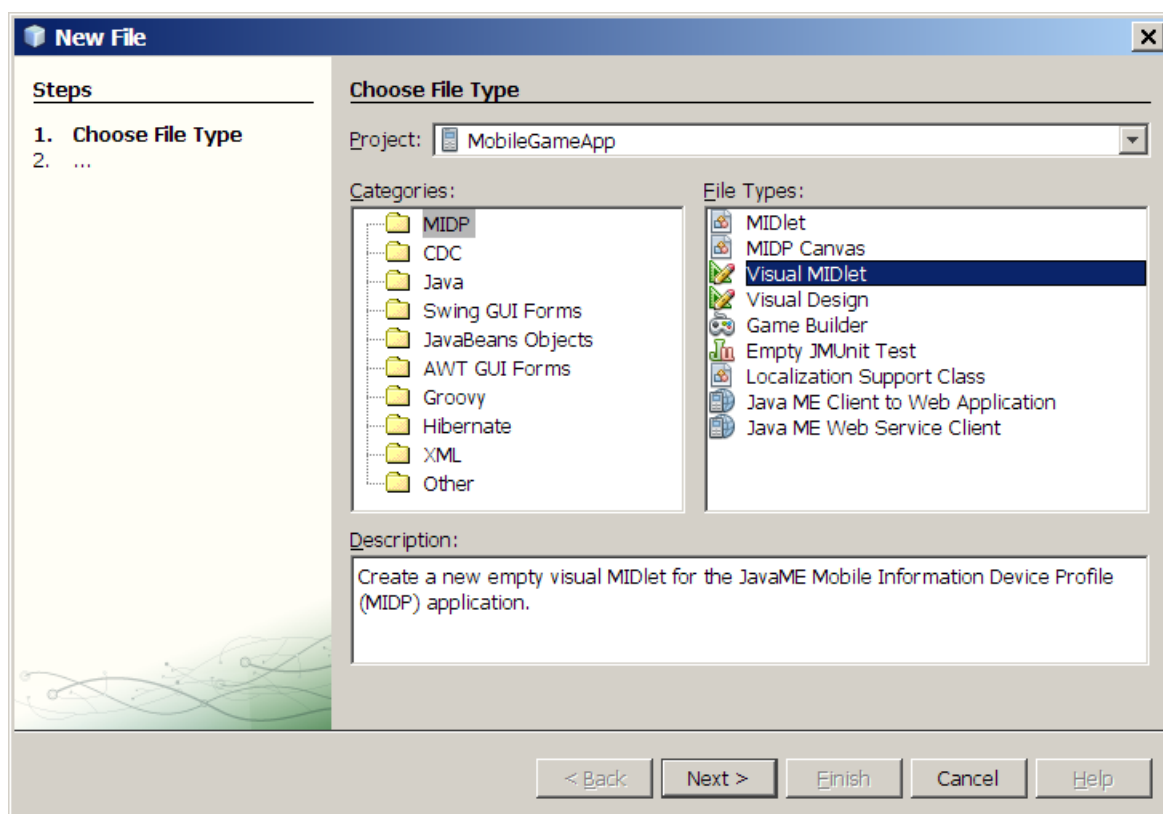
Mobilās spēle izstrāde var tikt veikta ar dažādiem rīkiem. Šajos rīkos ietilpst integrētā izstrādes vide, grafiskie redaktori, teksta redaktori. Šajā nodaļā tiek pārādītas dažas spēles izstrādes posmi, kas veicami ar integrēto izstrādes vidi NetBeans, grafisko redaktoru GIMP.

Integrēta izstrādes vide NetBeans piedāvā visplašākās iespējas spēles radīšanai tādas kā vizuālo mobilo izstrādātāju (*Visual Mobile Designer*), Mobilo spēļu izstrādātāju, kas ļauj rādīt spēles līmeņus, posmus grafiskā veidā. Tiek aprakstītas Java 2 Micro Edition klases, kas iekļautas spēles veidošanā, tiek parādīta izstrādāta mobila spēle lietotāja ceļvedī.

#### 3.1 Mobilas spēles radīšana ar NetBeans izstrādes vidi

Integrētā izstrādes vide NetBeans piedāvā plašas iespējas vizuālā Midleta izstrādāšanā. Izveidojot jauno vizuālā midleta failu (*Visual MIDlet*), paverās daudzi vizuālās izstrādes rīki.

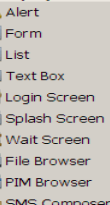
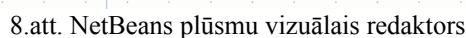
Izveidojot jauno vizuālo Midleta failu, parādās logs, kur piedāvāts izvēloties faila tipu “Visual midlet”. (attēls 7)



7.att. NetBeans MIDP failu tipi

Izvēloties jauno faila tipu parādās NetBeans izstrādes vides galvenais logs, kurā uzskaitīti vizuālie izstrādes rīki. Rīku skaitā analizētājs (*Analyzer*), Plūsmu diagrammu (*Flow*) vizuālais redaktors, Ekrāna (*screen*) vizuālais attēlotājs, kā arī pirmkoda redaktors

Plūsmu diagrammu redaktors – ir vizuālais izstrādes rīks(sk. 8 attēlu), kas ļauj pārnest komponentes no Paletes (*Palette*) (sk. Attēlu 9) un savienot tos.



**Palette**

- Databinding**
- Displayables**
  - Alert
  - Form
  - List
  - Text Box
  - Login Screen
  - Splash Screen
  - Wait Screen
  - File Browser
  - PIM Browser
  - SMS Composer
  - Commands**
    - Back Command

**splashScreen [SplashScreen] - Prop...**

**Properties**

|                         |                                     |
|-------------------------|-------------------------------------|
| Allow Timeout Interrupt | <input checked="" type="checkbox"/> |
| Full Screen             | <input type="checkbox"/>            |
| <b>Image</b>            | image1                              |
| <b>Text</b>             | Super Ronja                         |
| Text Font               | <None>                              |
| <b>Ticker</b>           | ticker                              |
| Timeout                 | 5000                                |
| <b>Title</b>            | Mobile game                         |
| <b>Code Properties</b>  |                                     |
| <b>Instance Name</b>    | splashScreen                        |
| Is Lazy Initialized     | <input checked="" type="checkbox"/> |

29

Cits vizuālais rīks ir Ekrāna (*Screen*) rīks, kas ļauj attēlot individuālā komponenta attēlojumu uz ekrāna, mainīt parametrus. (sk .attēlu 10)



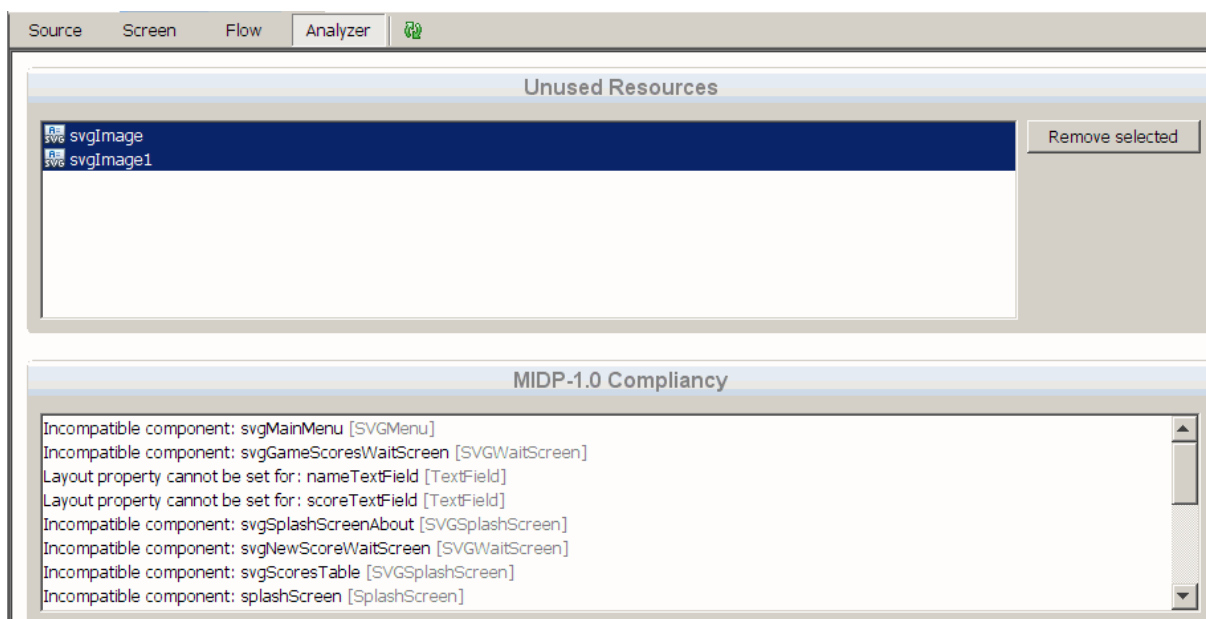
10. att. Ekrāna vizuālais rīks

Šajā rīkā var izvēlēties dažādus pieejamos resursus priekš dotā komponenta un paskarīties kā tas izskatīsies uz ekrānā, nepalaižot emulātoru. Vizuālie rīki automātiski ģenerē attiecīgo pirmkoda daļu. Uz uzplaiksnījuma ekrāna (*SplashScreen* )instances tiek pāradīts ģenerētais pirmkods, kas atbilst attēlam 10. “

```
public SplashScreen getSplashScreen() {
    if (splashScreen == null) {
        // write pre-init user code here
        splashScreen = new SplashScreen(getDisplay());
        splashScreen.setTitle("Mobile game");
        splashScreen.setTicker(getTicker());
        splashScreen.setCommandListener(this);
        splashScreen.setImage(getImage1());
        splashScreen.setText("Super Ronja");
        // write post-init user code here
    }
    return splashScreen;
}
```

Tiek redzams ģenerētais kods, kur uzplaiksnījuma ekrāns (*SplashScreen* )tiek inicializēts un tam tika piešķirti parametri, kas redzami attēlā nr. 9.

Analizētāja rīks, kas redzams uz 7.attēlā palīdz pārbaudīt savienojamību ar MIDP 1.0 profilu, pārbaudīt neizmantojamos resursus. (sk. Attēlu nr. 11).



11. att. NetBeans analizētājs vizuālam failam

Balstoties uz iepriekš parādītajiem piemēriem var secināt, ka NetBeans ļoti labi der vienkāršai mobilo lietojumu izstrādei, ar saviem vizuāliem rīkiem un komponentēm. NetBeans piedāvā plašas iespējas izstrādei un iespējas momentānai vizualizācijai.

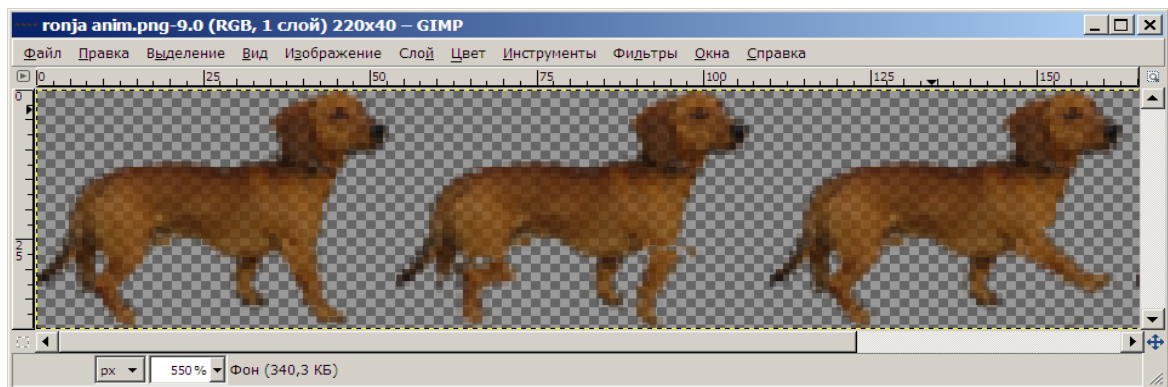
### 3.2 NetBeans vizuālie rīki spēles līmeņu izstrādei

Lai izstrādātu spēles NetBeans izstrādes videi ir rīks – GameBuilder. Ar to palīdzību var izveidot spēles laukumu(Scene), izveidot personāžu animāciju, ielādēt grafiskos failus.

Lai izmantotu GameBuilder jārada jauns fails ar tipu GameBuilder (sk. Attēlu 7).

GameBuilder rīks ļauj izstrādāt Spraitus, kā arī slāņus (*tiled layer*) .

Spraiti – divdimensiju animētais attēls, kas integrēts kopējā ainā. (*scene*). *Tiled Layer* – vizuālais elements, kas veidots ar šūnu laukumu. Ļauj veidot lielus spēļu laukumus, kuri sastāv no nelieliem objektiem(spraitiem). Spraiti priekš mobilās spēles ir grafiskais fails formātā \*.PNG (*Portable Network Graphics*). Spraitam ir jābūt caurspīdīgām. Spraiti var saturēt animāciju, animācija veidota tādā veidā, ka viens attēls var saturēt dažādus objektus. Kā piemēru, var parādīt grafiskā redaktorā GIMP izveidoto animēto spraitu. (sk. Attēlu 12)



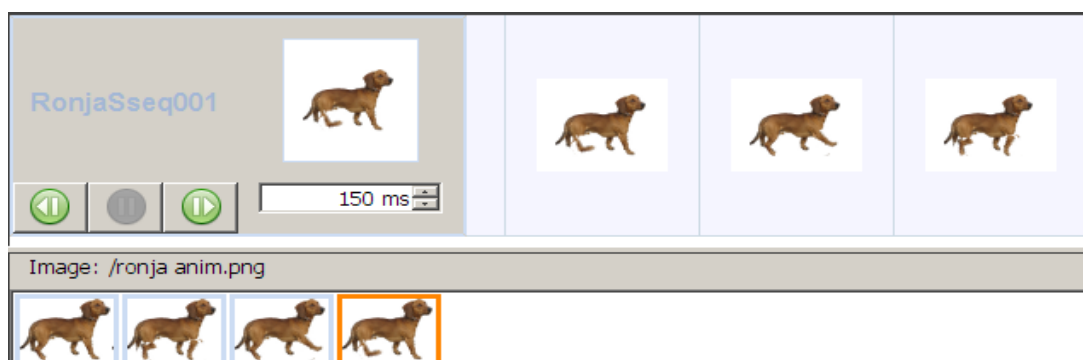
12.att. GIMP grafiskā redaktora radītais spraits

Izvēloties spraitu GameBuilder rīks prasa norādīt attēla parametrus, lai atdalīt spraita animācijas. Pēc šīs procedūras veikšanas tiek piedāvāts izveidot spraitam animāciju no iegūtiem atsevišķiem kadriem. (sk. 13 attēlu)

Tiek ģenerēts atbilstošais kods: “

```
public Image getRonja_anim() throws java.io.IOException {
    if (ronja_anim == null) {
        ronja_anim = Image.createImage("/ronja
anim.png");
    }
    return this.ronja_anim;
}

public Sprite getRonjaS() throws java.io.IOException {
    if (RonjaS == null) {
        RonjaS = new Sprite(getRonja_anim(), 55, 40);
        RonjaS.setFrameSequence(RonjaSseq001);
    }
    return RonjaS;
} “
```

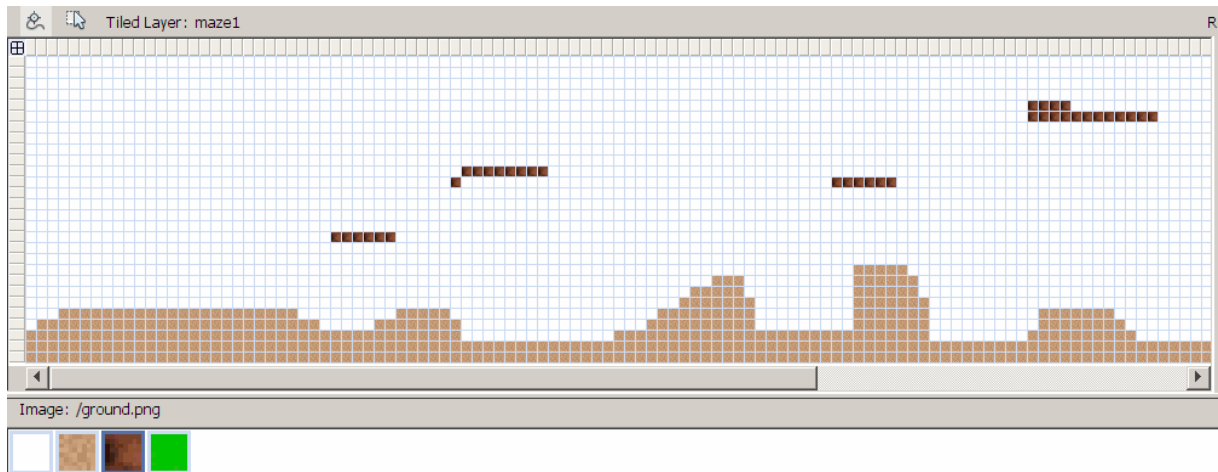


13. att. spraita animēšanas iespējas NetBeans GameBuilder rīkā

Radot *TiledLayer*, jāizvēlas attēls tādā pašā secībā ka spraitam. Pēc tam rīks izvadīs logu ar šūnām, kuras piedāvāts aizpildīt lai iegūtu domāto rezultātu. Piemērs *tiled layer* – mazel (sk. 14. attēlu).

Uz šo šūnu pamatā pirmtekstā ģenerēts masīvs pirmtekstā:





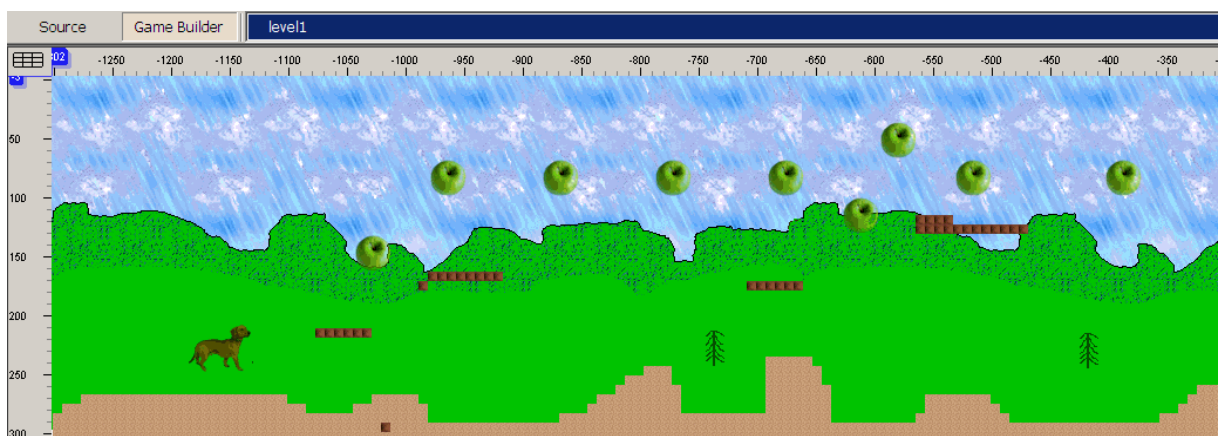
14. att. Tiled Layer maze1 radīšana

```
public TiledLayer getMaze1() throws java.io.IOException {
    if (maze1 == null) {
        // write pre-init user code here
        maze1 = new TiledLayer(161, 28, getGround(), 8,
8);

        int[][] tiles = {
            { 0, 0, 0, 0, 0, 0, (...)
            0, 0, 0, 0, 0, 0, (...) } }
```

Kur masīva elementi var būt 1,2,3 – attiecībā no izvēlētajā attēlā. 0 – nav izvēlēts elements.

Lai savienotu visus slāņus, spraitus vienā spēles ainā, jāizvēlas *GameBulider* rīkā “Scene”, un jāpievieno visi slāņi, spraiti. Pievienošana dod pilnīgu brīvību, tā kā tiek izmantota grafiskā izvēle un attēli var tikt brīvi pārvietoti, pa iedomāto spēles laukumu. Atsevišķi katru slāņi var “iesaldēt”, lai to nevarētu pēc tam pārvietot. Piemērā parādīta izstrādātā spēles aina. (sk. Attēlu 15).



15. att. Spēles ainas izveide ar GameBulider rīku

Pirmkodā līmenī tiek ģenerēts atbilstošais kods:

```
public void updateLayerManagerForLevel1(LayerManager lm)
throws java.io.IOException {
```

```

// write pre-update user code here
getApple_layer().setPosition(-1044, 35);
getApple_layer().setVisible(true);
lm.append(getApple_layer());
getRonjaS().setPosition(-1185, 208);
getRonjaS().setVisible(true);
lm.append(getRonjaS());
getMazel().setPosition(-1301, 83);
getMazel().setVisible(true);
lm.append(getMazel());
getJamesS().setPosition(-129, 166);
getJamesS().setVisible(true);
lm.append(getJamesS());
getTree1().setPosition(-425, 213);
getTree1().setVisible(true);
lm.append(getTree1());
getTr2().setPosition(-744, 211);
getTr2().setVisible(true);
lm.append(getTr2());
getTerra().setPosition(-1302, -3);
getTerra().setVisible(true);
lm.append(getTerra());
// write post-update user code here
}

```

Pēc visu rīku izmantošanas var teikt, ka NetBeans piedāvā ērtus vizuālos izstrādes rīkus, kas ļoti piemērotas mobilo spēļu radīšanu. Tie ir ērti tāpēc, ka grafiskie rīki ļoti atvieglo spēles pirmkoda rakstīšanu ģenerējot noteikto kodu un masīvs, ļauj apskatīt dažus elementus nepalaižot mobilās ierīces emulāciju. GameBulider ļauj viegli veidot spraitus un to animāciju no grafiskiem failiem. Spēles izstrādātajām jārūpējas tikai par spēles loģikas programmēšanu. Ja nebūtu šo rīku, programmētājam būtu ar rokām aizpildīt masīvus, importēt grafiskos failus, kas prasa ilgu laiku.

### 3.3 Java 2 ME klases lietošana spēles izstrādāšanā

Java2ME platforma piedāvā ļoti daudz spēļu programmēšanas iespējas. Tiek piedāvātas klases. Pakete `javax.microedition.lcdui.game` piedāvā daudz klases priekš spēles radīšanas. Klašu apraksts un iespējas piedāvātas tabulā 7. [Горняков, 2004]

7. tabula paketes `javax.microedition.lcdui.game` klases

| Klase      | Apraksts   |
|------------|--|
| GameCanvas | <p>Abstrakta klase GameCanvas atbild par ekrāna pārzīmēšanu, neizmantojot ieejas sistēmas pavedienus</p> <p>Klases GameCanvas metodes:</p> <ul style="list-style-type: none"> <li>void flushgraphics() - kopē attēlu no starpekrāna bufera ekrānā</li> </ul> |

|            |  |
|------------|--|
|            | <ul style="list-style-type: none"> <li>• void flushGraphics(int x, int y, int width, int height) – kopē attēlu no starpekrāna bufera un noteikto taisnstūri</li> <li>• protected Graphics getGraphics() - noteic grafiskos elementus lai attēlotu tos klasē GameCanvas</li> <li>• int getKeyStates() - noteic kura poga ir nospiesta</li> <li>• void paint(Graphics g) – zīmē grafiskos elementus, kuri attēloti klasē GameCanvas</li> </ul>   |
| Layer      | <p>Klase Layer norāda visas īpašības radītajiem spēles grafiskajiem slāņiem. Ar klases Layer var veidot slāņa izmēru un līmeņa pozīciju</p> <ul style="list-style-type: none"> <li>• int getHeight() - saņem ekrāna augstumu</li> <li>• int getWidth() - saņem ekrāna platumu</li> <li>• int getX() - saņem horizontālo slāņa koordināti</li> <li>• int getY() - saņem vertikālo slāņa koordināti</li> <li>• void move(int dx, int dy) – pārvieto slāņi uz dx un dy koordinātēm</li> <li>• abstract void paint(Graphics g) – zīmē slāņi</li> <li>• void setPosition(int x, int y) nosaka slāņa pozīciju, kas apzīmējās ar x un y koordinātēm</li> </ul>  |
| TiledLayer | <p>Zīmē spēles fonu attēlu. Fona attēls ir zīmēts šūnu veidā. Šūnu skaits un izvietojums var būt patvaļīgs. Kases TiledLayer konstruktors:<br/> TiledLayer(int columns, int rows, Image image, int tileWidth, int tileHeight) , kur columns – kolonnu skaits<br/> rows – rindu skaits<br/> image – attēls<br/> tileWidth – šūnas platumas izmērs pikseļos<br/> tileHeight – šūnas augstuma izmērs pikseļos<br/> Klases TiledLayer metodes:</p> <ul style="list-style-type: none"> <li>• int createAnimatedTile(int staticTileIndex) – rada animācijas fonu un atgriež nākamās šūnas indeksu</li> <li>• int getCellHeight() - saņem šūnas augstumu pikseļos</li> <li>• int getCellWidth() - saņem šūnas platumu</li> <li>• int getColumns() - saņem kolonnu skaitu</li> </ul> |

|              |  |
|--------------|--|
|              | <ul style="list-style-type: none"> <li>• int getRows() - saņem rindu skaitu</li> <li>• void paint() - zīmē fonu</li> <li>• void setCell(int col,int row, int tileIndex) – zīmē atbilstošu šūnu</li> </ul>  |
| LayerManager | <p>Klase LayerManager atbalsta jebkura grafiska objekta attēlojumu spēles ainā. Lai radītu objektu, vajag izmantot klases konstruktoru:<br/> LayerManager()<br/> Piemērs LayerManager l new LayerManager();<br/> Klases LayerManager metodes:</p> <ul style="list-style-type: none"> <li>• void append(Layer l) – pievieno slāņi l līmeņu menedžerā</li> <li>• Layer getLayetAt(int index) – iegūst līmeņi ar noteikto indeksu</li> <li>• int getSize() - iegūst līmeņu kopējo skaitu</li> <li>• void insert(Layer l, int x, int y) – noteic jauno līmeņi uz noteikto indeksu</li> <li>• void paint(Graphics g, int x, int y) – atbalsta tekošo līmeņu menedzeri noteiktajās koordinātēs</li> <li>• void remove(Layer l) – dzēš līmeni</li> </ul>  |
| Sprite       | <p>Klase Sprite ir līdzīga klasei TiledLayer. Bet tā darbojas ar vienu attēlu, kas veidu kādu spēles personāžu, nevis fona attēlu. Ielādējamaais attēls var tikt animēts. Animāciju skaits ir neierobežots. Attēla kadram jābūt vienāda garuma. Klasei Sprite piemīt 3 konstruktori:</p> <ul style="list-style-type: none"> <li>• Sprite (Image image) – rada neanimēto spraitu</li> <li>• Sprite(Image image, int frameWithd, int frameHeight) – rada animēto spraitu, no attēla kadriem.</li> <li>• Sprite(Sprite s) – veido spraitu no cita spraita</li> </ul> <p>Klases Sprite metodes:</p> <ul style="list-style-type: none"> <li>• boolean collidesWith(Sprite s, boolean pixelLevel) – noteic divu spraitu sadursmi</li> <li>• boolean collidesWith(TiledLayer t, boolean pixelLevel) – noteic spraita sadursmi ar Klases TiledLAYER robežu</li> <li>• int getFrame() - iegūst tekošo kadru</li> <li>• void nextFrame() - pāreja uz nākamo kadru</li> </ul> |

|  |   |
|--|---|
|  | <ul style="list-style-type: none"> <li>• void prevFrame() - pāreja uz iepriekšējo kadru</li> <li>• paint (Graphics g) – zīmē spraitu</li> <li>• void setFrame(int sequenceIndex) – rada noteikto kadru</li> <li>• void setFrameSequence(int[] sequence) – noteic noteiktu kadru secību</li> <li>• void setImage(Image img, int frameWidth, int frameHeight) – aizvieto spraita attēlu</li> <li>• void setTransform(int transform) – noteic Spraita transformāciju</li> <li>• public void defineReferencePixel(int x, int y) – izmaina spraita policiju, pārnesot to koordinātēs x, y</li> <li>• setTransform() - transformē spraitu apgriežot to uz noteikto leņķi, konstantes: <ul style="list-style-type: none"> <li>◦ static int TRANS_MIRROR</li> <li>◦ static int TRANS_MIRROR_ROT180</li> <li>◦ static int TRANS_MIRROR_ROT_270</li> <li>◦ static int TRANS_MIRROR_ROT_90</li> <li>◦ static int TRANS_NONE</li> <li>◦ static int TRANS_ROT180</li> <li>◦ static int TRANS_ROT270</li> <li>◦ static int TRANS_ROT90</li> </ul> </li> </ul> |
|--|---|

Pēc tabulas izskatīšanas redzams, kā paketes `javax.microedition.lcdui.game` klases palīdz ērti veidot mobilās spēles elementus un veikt ar tām dažādas transformācijas. Klases ir neatkarīgas un to izmantošana ļauj pietiekami labi uzrakstīt spēles ainu ar spēles elementiem. Šīs klases ir ļoti noderīgas lai aprakstītu spēles elementu mijiedarbību.

### 3.4 Izstrādātas spēles apraksts

Izveidotais mobilais lietojums ir izstrādāts priekš *CLDC* konfigurācijas un *MIDP 2.0* profila atbalstītajiem lietojumiem. Lietojums izstrādāts NetBeans 6.5 integrētā izstrādes vidē ar spēles izstrādes šablona pielietošanu.

### 3.4.1 Spēles izstrādes metodoloģijas apraksts

Nelielajiem projektiem ir ļoti noderīga Open Source izstrādes spējas metodoloģija. Open Source metodoloģijai nav īpašu definēto izstrādes fāžu, tā ļauj pilnīgu brīvu izstrādi.

Tekošā mobilās spēles izstrādē tiek izvēlēta Open Source izstrādes metodoloģija.

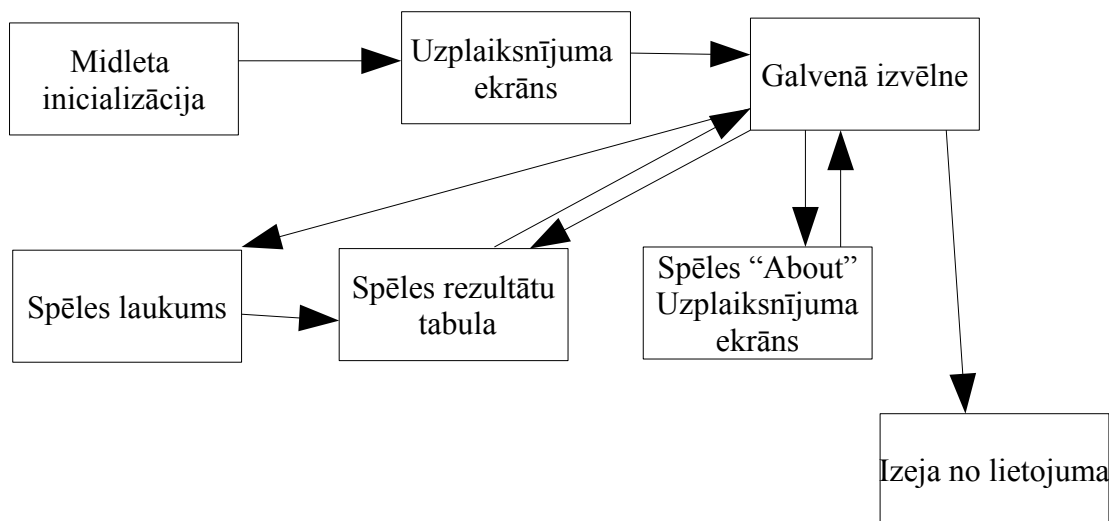
Open Source īpašības:[Nikiforova 1]

- ☐ Programmas kods ir vienīga dokumentācija
- ☐ Projekta koordinators pārskata izstrādātāju piedāvājumus un ieraksta koda repozitorijā
- ☐ Projekts var būt sadalīts moduļos, tad katram modulim savs koordinators
- ☐ Lielākā daļa tiek tērēta kļūdu meklēšanai un labošanai

Open Source izstrādes metodoloģijai ir priekšrocība, kā tā ir noderīga mazos projektos, kur nepieciešama radoša noskaņa. Visas nepieciešamas piezīmes tiek ierakstītas pirmkodā.

### 3.4.2 Spēles struktūra

Spēle sastāv no vairākām daļām: palaišanas daļa, uzplaiksnījuma ekrāna, galvenās izvēlnes, mobilās spēles apstrādes daļas. Attēlā 16 tiek parādīta šo elementu mijiedarbības diagramma.



16. att. Lietojuma diagramma

### 3.4.3 Spēles pirmkoda apraksts

Liela daļa no lietojuma koda tika ģenerēta ar NetBeans izstrādes vidi. Tāpēc šajā apakšnodaļā tiks apskatīta daļa no spēles pirmkoda (metodes), kas atbild par spēles loģiku un spēles ainas elementu mijiedarbība. Ģenerētas pirmkoda metodes netiks apskatītas. Ar

pirmkoda fragmentu (klase MazeManager) var iepazīties 2 pielikumā.

Spēles pirmkoda faili ir sadalīti 2 paketēs:

- org.netbeans.j1.game
- org.netbeans.j1.game.logic

Paketes org.netbeans.j1.game klašu hierarhija (iekļaujot pamatklares) :

java.lang.Object

- javax.microedition.lcdui.Displayable
  - javax.microedition.lcdui.Canvas
    - javax.microedition.lcdui.game.GameCanvas
      - org.netbeans.j1.game.**MazeCanvas**- org.netbeans.j1.game.**MazeGameDesign**
- javax.microedition.midlet.MIDlet
  - org.netbeans.j1.game.**GameMIDlet**

Paketes org.netbeans.j1.game.logic klašu hierarhija:

- java.lang.Object
  - javax.microedition.lcdui.game.LayerManager
    - org.netbeans.j1.game.logic.**MazeManager**
  - java.lang.Thread (implements java.lang.Runnable)
    - org.netbeans.j1.game.logic.**GameThread**

Tādejādi redzams kā pakešu klases ir noteikto pamatklašu atvasinājumi.

Klases GameMIDlet metodes:

- public void **gameOver**(int score)

Aptur spēles ainas pavadīšanu, kad izsaukts un izsauc spēles punktu metodi.

- public void **svgMainMenuAction**()

Veic darbības kas piešķirtās attiecīgām SVG izvēlnes elementiem svgMainMenu komponentē.

- public void **exitMIDlet**()

Veic izeju no lietojuma

- public void **startApp**()

Tiek izsaukts, kad Midlets tiek palaists. Pārbauda vai Midlets ir palaists un inicializē vai atjauno to.

- public void **pauseApp**()

Tiek izsaukts, kad Midlets ir pauzes režīmā

protected void **updateSvgWithHighScores()**

Atjauno SVG tabulu ar augstākajiem spēles punktiem

Klases MazeCanvas metodes:

- void **checkKeys** ()  
Atbild uz taustiņu nospiedumiem.
- void **flushKeys** ()  
Attīra taustiņu buferi.
- MazeManager **getManager** ()  
Dabū MazeManager.
- void **keyPressed** (int keyCode)  
Atbild uz taustiņu nospiedumiem.
- void **paint** (javax.microedition.lcdui.Graphics g)  
Zīmē grafiku uz ekrāna – spēlētāja nosaukumu – Ronja un punktu skaitu - “Scores”.  
Ja spēle beidzās zīmē zilo kvadrātu au uzrakstu par spēles beigšanu.
- void **reset** ()  
Izdala visus mainīgos to sākotnējā stāvoklī.
- void **setGameOver** ()  
Tiek izsaukts, kad beidzas spēle.
- void **start** ()  
Metode ielādējās un inicializē grafiskos objektus priekš taimera.
- void **updateScreen** ()  
Atjauno ekrānu.

Paketes org.netbeans.j1.game.logic apskatāmas klases

**GameThread** Klase satur galveno ciklu, kas nodrošina spēles gaitu.

**MazeManager** Klase satur galveno ciklu, kas nodrošina spēles gaitu.

Standarta klases GameThread konstruktors:

**GameThread** (MazeCanvas canvas, GameMIDlet dungeon)

Klases GameThread metodes:

- private long **getWaitTime** ()  
Nosaka gaidīšanas laiku kas jāveic pirms nākamās ekrāna atjaunošanas, lai ierobežotu kadru skaitu un taupītu resursus



- `void pause()`  
apstādina spēli
- `void requestStop()`  
pārtrauc spēles ainu
- `void run()`  
Sāka spēli  
Klases `MazeManager` metodes:
- `void init()`  
Inicializē visus datus
- `void paint(javax.microedition.lcdui.Graphics g)`  
Zīmē spēles grafiku uz ekrāna
- `void requestMove(int hdirection, int vdirection)`  
Atbild uz noteiktu taustiņu nospiedumiem un noteic kur pārvietot spēles personāžus objektus.
- `void reset()`  
Attīra visus mainīgos
- `private void collapple()`  
Pārbauda vai ir notikusi sadursmi ar kādu no ābolu spraitiem, jā notikusi paslēpj attiecīgā ābola spraitu un uzstāda attiecīgo punktu skaitu
- `private void jmove()`  
Pārvieto Lielā Apļa spraitu uz augšu un atpakaļ
- `boolean checkCollision()`  
Pārbauda spēlētāja personāža spraita sadursmi ar grīdu
- `private void updateSprite(int hdirection, int vdirection)`  
Nosaka spēlētāja personāža spraita izskatu, atkarībā no pārvietoējuma.

### 3.5 Izstrādāta JAR faila struktūra

Izstrādes beigās tiek rādīts nokompilētais JAR un JAD fails. Attiecīgie failu nosaukumi: `SuperRonja.JAD` un `SuperRonja.JAR`.

Fails `SuperRonja.JAD` satur Midleta atpakstu, lai instalētu to mobilajā ierīcē.

Faila `SuperRonja.JAR` struktūra attēlota 17 attēlā. Saknes direktorijā ir 2 direktorijas un 12 grafiskie faili. Grafiskie faili tiek izmantoti, mobilā spēlēs ainā, kā arī uzplaiksnījuma ekrānos.

| Name                  | Size   | Packed Size |
|-----------------------|--------|-------------|
| org                   | 353 K  | 144 K       |
| META-INF              | 309 B  | 210 B       |
| 1terrain.png          | 235 K  | 235 K       |
| about.png             | 108 K  | 108 K       |
| scoreTable.svg        | 31 K   | 12 K        |
| waitscreen.svg        | 29 K   | 10 K        |
| ronja_splash2.jpg     | 17 K   | 16 K        |
| menu.svg              | 11 K   | 3836 B      |
| circle1.png           | 7055 B | 7060 B      |
| ronja_anim.png        | 5437 B | 5442 B      |
| apple_green_fruit.png | 2735 B | 2740 B      |
| red_a.png             | 2541 B | 2546 B      |
| ground.png            | 489 B  | 494 B       |
| tree.png              | 389 B  | 387 B       |

17. att. Faila SuperRonja.JAR struktūra

Direktorijā META-INF satur failu MANIFEST.MF, kas apraksta mobilo lietojumu.

Direktorijā org\\netbeans\\ atrodas 2 direktorijas:

- microedition – satur iekļautās paketes un bibliotēkas, kas nepieciešamas mobilajiem lietojumam
- j1 – satur izstrādātus nokompilētus pirmkoda failus, gatavus izpildīšanai (skatītiem 18 attēlu)

| Name                 | Size   | Packed Size |
|----------------------|--------|-------------|
| logic                | 10 K   | 5421 B      |
| MazeGameDesign.class | 32 K   | 4277 B      |
| GameMIDlet.class     | 13 K   | 5898 B      |
| MazeCanvas.class     | 5148 B | 2514 B      |
| GameMIDlet\$2.class  | 1011 B | 545 B       |
| GameMIDlet\$1.class  | 886 B  | 489 B       |
| GameMIDlet\$3.class  | 690 B  | 416 B       |

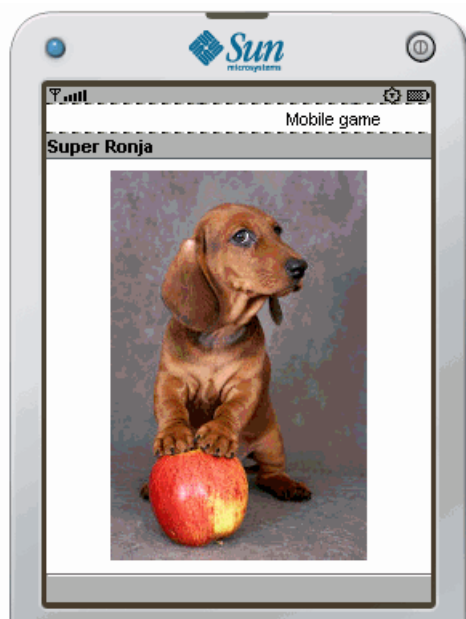
18. att. Faila SuperRonja.JAR \\org\\netbeans\\j1\\game\\ direktorija

### 3.6 Lietotāja ceļvedis

Mobilā spēle SuperRonja apskatāma pie Darbības/Platformu (*Action/Platformer*) spēļu žanra. Mobila spēle izpildīsies uz J2ME tehnoloģijas MIDP 2.0 atbalstītām ierīcēm (vairākums no 2005 gada izlaistiem mobiliem telefoniem un plaukstdatoriem). Lai instalētu spēli mobilajā ierīcē, jānokopē faili SuperRonja.JAD un SuperRonja.JAR attiecīgajā direktorijā mobilajā ierīcē. (skatīties savas mobilās ierīces lietotāja ceļvedi). Instalējot spēli Ja uz ekrāna parādās uzraksts “*Not a signed application. Continue Download?*” jānospiež “Yes” poga. Pēc instalēšanas jūsu mobilajā ierīcē parādīsies mobilais lietojums ar nosaukumu :SuperRonja.

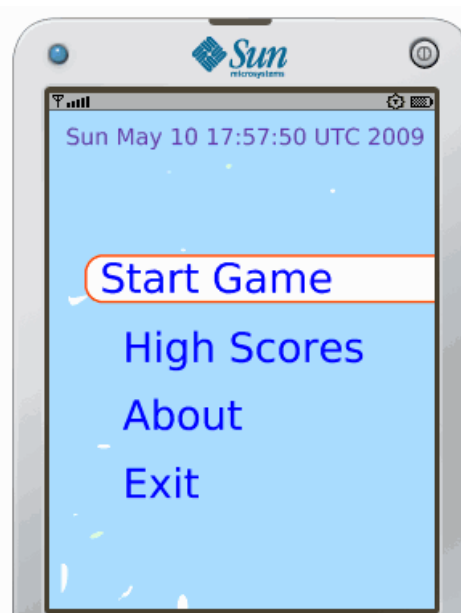
Pēc mobilā lietojuma izsaukšanas parādīsies uzplaiksnījuma ekrāns ar spēles logotipu

un spēles nosaukumu. (sk. 19 attēlu)



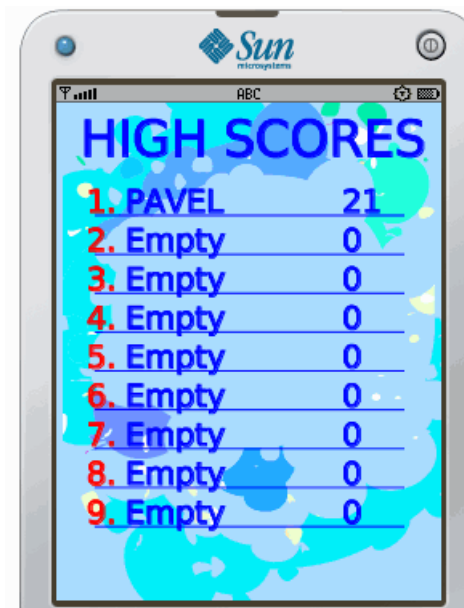
19. att. spēles uzplaiksnījuma ekrāns

Pēc uzplaiksnījuma ekrāna parādās spēles izvēlne ar piedāvātām iespējām. Spēles izvēlne ir angļu valodā. Izvēlnes saturs: “*Start Game, High Scores, About, Exit*”. (skatīties 20. attēlu)



20. att. Spēles izvēlne

Izvēloties “*Start Game*” punktu tiek uzsākta spēle. Vēlāk tiks apskatīta spēles gaita. Izvēloties izvēlni “*High Scores*” tiek parādītā labāko spēlētāju tabula. (skatīties 21 attēlu).



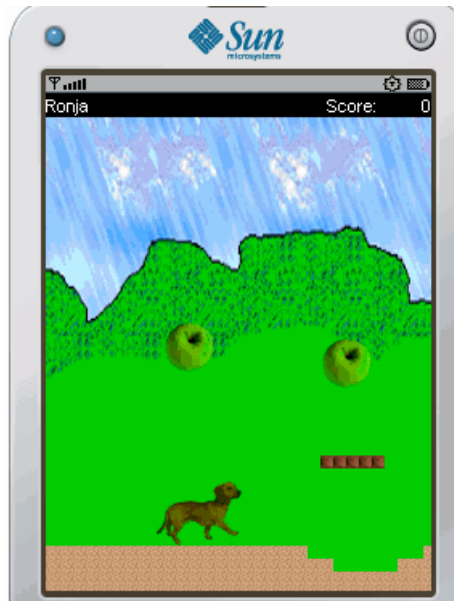
21.att. Spēles labāko rezultātu tabula

Izvēloties izvēlni “About” tiek parādīts uzplaiksnījuma ekrāns ar spēles nosaukumu, noteikumiem un izstrādātāju. Instrukcija: “Jāņem zaļie āboli, tad jāņem sarkanais ābols, kas reizinās jūsu punktus ar 3. Lai pabeigtu spēli jāiet pie Lielā Apļa.” (Skatīties 22 attēlu).



22. att. spēles instrukcija

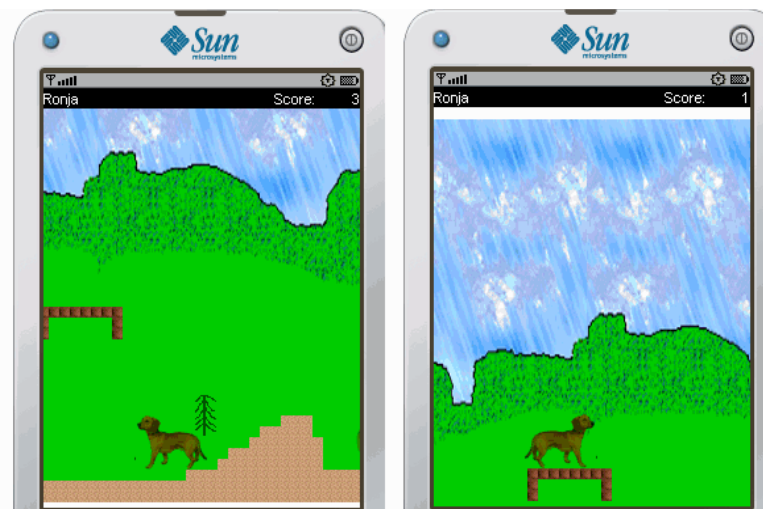
Spēles uzsākšana: Uzsākot spēli lietotājs redz savu spēles personāžu – suns (taksis), ābolus, grīdu un platformas. (skatīties 23 attēlu)



23. att. Spēles sākums

Augšējā kreisajā ekrānā daļā lietotājs redz sava personāža vārdu - “Ronja” un labajā daļā punktu skaitu.

Spēles personāžs var virzīties divos virzienos: pa labi un pa kreisi, attiecīgi spiežot labo un kreiso mobilo ierīces taustiņu. (Skatīties 24. attēlu) Lai izdarītu lēcieni un uzlektu uz platformas spēlētājam jānospiež augšējo taustiņu vai augšējo + labo vai kreiso taustiņu, lai lektu attiecīgajā virzienā. Lai savāktu vairāk spēles punktus jāņem sarkanais ābols, kas reizina spēles punktu skaitu ar 3. (Skatīties 25 attēlu)



24.att. iešana pa labi vai pa kreisi

Lai pabeigtu spēli jāpieiet pie Lielā apļa, tad beigsies spēle, pārārdīsies zilais logs ar uzrakstu “Game Over” un punktu skaitu. (Skatīties 26. attēlu)



25. att. Spēles punkti pirms un pēc sarkana ābola paņemšanas



26. att. Lielais aplis un spēles beigas

Pēc spēles beigām tiks pieprasīts ievadīt spēlētāja vārdu un pēc vārda ievades tiks pārārdīta labāko spēlētāju tabula(sk. 21 attēlu). Pēc tabulas parādīšanas spēle atgriezīsies galvenajā izvēlnē. Iziet no spēles var izvēloties izvēli “Exit” (sk. 20 attēlu).

## SECINĀJUMI

Bakalaura darba ietvaros izpētīta Java ME tehnoloģija. Aprakstīti Java ME tehnoloģijas komponentes un tika izveidoti secinājumi par tiem. Darbā dots priekšstats par mobilajām spēlēm, to popularitāti un to tirgu apjomu.

Pētījuma ietvaros izstrādāta spēle ar nosaukumu “Super Ronja”, kas pieder pie darbības (*Action*) spēles žanra. Spēlēs mērķis parādīt, kā ar NetBeans integrētās izstrādes vides rīkiem var ļoti vienkārši izveidot mobilo spēli.

Java 2 ME tehnoloģija atradusi savu pielietojamību daudzās mobilajās ierīcēs. Galvenokārt to izmanto mobilos tālruņos un kabatas datoros. Java 2 ME platforma ir optimizēta priekš mobilajām ierīcēm ar ievērojamāki mazākajiem resursiem, salīdzinājumā personālie datori. Ar Java 2 ME tehnoloģijas palīdzību var izveidot pilnvērtīgus mobilos lietojumus, īpaši izmantojot MID profilu 2.0.

Arī tika salīdzinātas integrētas izstrādes vides NetBeans un EclipseME, kā arī Sun Wireless toolkit rīku īpašības. Veiktais pētījums uzrādīja, kā NetBeans integrēta izstrādes vide ļauj pilnīgi realizēt spēļu izstrādāšanās potenciālu, piedāvājot vairākus vizuālos izstrādes rīkus. NetBeans piedāvā ērtus lietojuma veidošanas rīkus, kā arī ļauj grafiskā veidā manipulēt ar spēļu atribūtiem, spraitiem, veidot spēles ainas. Visu paveikto grafisko darbību paveikšanas automātiski tika ģenerēts attiecīgais pirmkods. Tādi rīki ļoti atvieglo programmētāja darbību. Programmētāja darbību arī atvieglo automātiska sintakses pārbaude.

Darbā organizācijas un rakstīšanas gaitā tika konstatēta nepietiekama grāmatu pieejamība. Tas tāpēc, ka Java ME tehnoloģija ir ļoti jauna un pagaidām Latvijā nav tik liels pieprasījums uz šādam grāmatām. Nācās vairāku materiāla daļu ņemt no interneta resursiem, tas arī var tikt uzskatāmi par priekšrocību, jo interneta resursos atspoguļojās vissvaigākā un visaktuālākā informācija, kas ir pieejama uz doto brīdi. Loti daudz informācijas atrodas uz Sun Microsystems, Java tehnoloģijas izstrādātāja, mājas lapas. Informācija ļoti palīdzēja darbā gaitā, ļauja pilnīgi apskatīt izvirzīto tēmu. Arī lietojot spēles izveidošanas terminus tika konstatēts latviešu valodas terminu iztrūkums (šo terminu vienkārši nav).

Pirms mobilas spēles izstrādes bija daudzas domas kā pilnvērtīgi izstrādāt mobilo spēli, tomēr pēc integrētas izstrādes vides analīzes un salīdzinājuma izvēle ir kritusi uz NetBeans vidi. NetBeans integrētā izstrādes vide piedāvā ļoti vienkāršus piemērus, kā veidot dažādus lietojumus, arī mobilas spēles. Pēc visu piemēru un aprakstu izskatīšanas tika nolemts izstrādāt mobilo spēli uz NetBeans piemēra.

Pēc darba pabeigšanas šī darba autors uzstata, kā darba mērķi un uzdevumi tika izpildīti.

Pētot šī darba tematiku, var izdarīt secinājumus, ka šī ir ļoti plaša tēma, ja ļaut plaši to aprakstīt, lapaspušu apjoms var pārsniegt 200 lapaspuses. Galvenais mērķis iepazīties ar Java 2 ME tehnoloģiju un veidot priekšstatu par mobilo spēļu izstrādi.

Šis darbs var noderēt kā īsa pamācība, lai izveidotu savu mobilo spēli, kā arī lai iepazīties ar Java ME tehnoloģiju un mobilo spēļu izstrādes aspektiem. Tajā ir iekļauta neliela daļa, kā veidot ar NetBeans mobilas spēles grafiskos elementus, kā arī nepieciešamo klašu apraksts spēļu izveidošanai.



# LITERATŪRA

[DSDP Mobile Tools ] DSDP Mobile Tools for Java Webinar, 2009 by Motorola and others/Internets - [http://www.eclipse.org/community/training/webinars/090422\\_MTJ\\_Webinar.pdf](http://www.eclipse.org/community/training/webinars/090422_MTJ_Webinar.pdf)

[ECLIPSE 1] EclipseME /Internets. - <http://eclipseme.org/docs/index.html>

[FINANCIAL 1] The 10-yr story: Java and the networked world/Internets. - <http://www.financialexpress.com/news/The-10-yr-story:-Java-and-the-networked-world/68607/>

[Jafar Ajdari 2001] Java 2 Mobile Information Device Profile (MIDP) Jafar Ajdari Helsinki University of Technology /Internets - [www.tml.tkk.fi/Studies/Tik-111.590/2001s/papers/jafar\\_ajdari.pdf](http://www.tml.tkk.fi/Studies/Tik-111.590/2001s/papers/jafar_ajdari.pdf)

[NETBEANS 1] NetBeans IDE - Java ME Development/Internets. - <http://www.netbeans.org/features/javame/index.html>

[Nikiforova 1] Oksana Nikiforova “Programatūras attīstības tehnoloģijas” - , 8 slaidi, 2008

[NIELSENMOBILE1] Nielsen Mobile: Press Release/Internets. - [http://www.nielsenmobile.com/html/GDC07\\_press\\_release\\_template.html](http://www.nielsenmobile.com/html/GDC07_press_release_template.html)

[PEABODY 1] The Art of Computer Game Design/Internets- <http://www.vancouver.wsu.edu/fac/peabody/game-book/Chapter3.html>

[ROBIN 1] Genre and the Video Game/Internets. - <http://www.robinlionheart.com/gamedev/genres.xhtml>

[SUN 1] About the Java Technology (The Java™ Tutorials > Getting Started > The Java Technology/Internets. - <http://java.sun.com/docs/books/tutorial/getStarted/intro/definition.html>

[SUN 2] About Java Technology/Internets <http://www.sun.com/java/about/>

[SUN 3] Java ME technology /Internets. - <http://java.sun.com/javame/technology/index.jsp>

[SUN 4] Connected Limited Device Configuration (CLDC); JSR 30, JSR 139 Overview/Internets. - <http://java.sun.com/products/cldc/overview.html>

[SUN 44] Mobile Information Device Profile/Internets. - <http://java.sun.com/products/midp/whatsnew.html>

[SUN 5] Mobile Information Device Profile (JSR-37) JCP Specification Java 2 Platform, Micro Edition, 1.0a, Sun Microsystems, Inc., 1.0a, December 15, 2000]

[SUN 6] Sun Java Wireless Toolkit for CLDC (formerly known as Java 2 Platform, Micro Edition (J2ME) Wireless Toolkit)/Internets. - <http://java.sun.com/products/sjwtoolkit/>

[SUN7] Jar Guide/Internets. - <http://java.sun.com/j2se/1.5.0/docs/guide/jar/jarGuide.html>

[SUN8] JAR File Specification/Internets. - <http://java.sun.com/j2se/1.5.0/docs/guide/jar/jar.html>

[TELEPHIA 1] Telephia: Your connection to the digital consumer/Internets. - [http://www.telephia.com/html/insights\\_071206.html](http://www.telephia.com/html/insights_071206.html)

[TELEPHIA 2] Mobile Gaming Stats/Internets. - <http://www.telephia.com/html/press%20releases/MobileGamingStats.html>

[TELEPHIA 3] Top Ten U.S. Lists 2008/Internets. - <http://www.telephia.com/html/press%20releases/TopTenU.S.Lists2008.html>

[Wikipedia 1] /Internets. - [http://en.wikipedia.org/wiki/Mobile\\_Information\\_Device\\_Profile](http://en.wikipedia.org/wiki/Mobile_Information_Device_Profile)

[Wikipedia 2] /Internets. - <http://en.wikipedia.org/wiki/MIDlet>

[WIKIPEDIA 3] Video game genres/Internets. - [http://en.wikipedia.org/wiki/Video\\_game\\_genres](http://en.wikipedia.org/wiki/Video_game_genres)

[Wikipedia4] JAR (file format)/Internets. - [http://en.wikipedia.org/wiki/JAR\\_%28file\\_format%29](http://en.wikipedia.org/wiki/JAR_%28file_format%29)

[Горнаков 2004] Горнаков С.Г. Программирование мобильных телефонов на Java 2 Micro Edition, ДМК-Пресс, 2004,-336с.

[Монахов 2008] В.Монахов, Язык программирования Java и среда NetBeans – БХВ-Петербург, 2008,-640с.

## **PIELIKUMI**

## 1. pielikums

### Mobilo spēļu žanru piemēri



Darbības spēle Tom and Jerry: Mouse Maze", GlibalFun



Darbības - cīņu spēle Prince of Persia HD - Gameloft



Loģiskā spēle Tetris Pop - EA Mobile



Sporta spēle FIFA 2009 - EA Mobile



Stratēģijas spēle Panzer Tactics 2 - HandyGames



Sacikšu spēle - Le Mans 2006 - Infospace

### Spēles loģikas pirmkoda fragments no faila MazeManager.java

```
/*
 * Šis pirmkoda fragments ir izplatīts brīvi, ar atsauci uz autoriem
 * Copyright (c) 2008, Lukas Hasik, Sun Microsystems, Inc.
 * Copyright (c) 2007, Carol Hamer, Apress
 * Copyright (c) 2009, Pavel Kartashev
 */

public MazeManager(int x, int y, int width, int height, MazeCanvas
canvas) {
    myCanvas = canvas;
    CANVAS_X = x;
    CANVAS_Y = y;
    DISP_WIDTH = width;
    DISP_HEIGHT = height;
} //klases konstruktors

/**
 * inicializē visus datus
 *
 * Atsevišķa metode init ielādē spēles datus, kad tie ir vajadzīgi
 */

public void init() throws Exception {

    mazeDesign = new MazeGameDesign(); //
    jamesSprite = mazeDesign.getJamesS();//
    RonjaSprite = mazeDesign.getRonjaS();//
    RonjaSprite.defineReferencePixel(RonjaSprite.getWidth() / 2,
0);//
    RonjaSpriteAnimator = new SpriteAnimationTask(RonjaSprite,
false);//
    myWalls = mazeDesign.getMaze1();//
    apple = mazeDesign.getApple();
    a2 = mazeDesign.getA2();
    a3 = mazeDesign.getA3();
    a4 = mazeDesign.getA4();
    a5 = mazeDesign.getA5();
    a6 = mazeDesign.getA6();
    a7 = mazeDesign.getA7();
    a8 = mazeDesign.getA8();
    a1 = mazeDesign.getRa();
    mazeDesign.updateLayerManagerForLevel1(this);//[Exercise3]

    timer = new Timer();//
    timer.scheduleAtFixedRate(RonjaSpriteAnimator, 0,
mazeDesign.RonjaSseq001Delay);//[Exercise3]

    //Nosaka, lai galvenais personažs būtu ekrāna lejupdaļa un centrā

    myViewWindowX = RonjaSprite.getX() - ((DISP_HEIGHT -
SQUARE_WIDTH) / 4);
    myViewWindowY = RonjaSprite.getY() - ((DISP_HEIGHT -
```

```

    SQUARE_WIDTH) -36);

        System.gc();
    }

    /**
     * atgriež visus mainīgos savās vietās
     */
    public void reset() {
        remove(jamesSprite);
        remove(RonjaSprite);
        remove(myWalls);
        remove(apple);
        RonjaSpriteAnimator.cancel();

        lastVertDirection = 0;
        lastHorDirection = 0;
        myIsRunning = 0;
        myIsJumping = NO_JUMP;

        System.gc();
    }

    //-----
    //  Grafiskas metodes
    /**
     * zīmē grafiku uz ekrāna
     */
    public void paint(Graphics g) throws Exception {
        g.setColor(MazeCanvas.WHITE);
        // aizkrāso ekrānu baltā krāsā, lai nokrāsotu nevajadzīgus
elementus

        g.fillRect(0, 0, DISP_WIDTH, DISP_HEIGHT);

        setViewWindow(myViewWindowX, myViewWindowY, DISP_WIDTH,
DISP_HEIGHT);
        // paint funkcijas izsaukums no superclases LayerManager
        // to paint all of the Layers
        paint(g, CANVAS_X, CANVAS_Y);
    }

    //-----
    //  Spēles personāža kustības
    /**
     * Atbild uz taustiņu nospiedumiem un nostāda attiecīgos spēles
objektus
     */
    public void requestMove(int hdirection, int vdirection) {
        // Identificē lēcieni, ja personāžs nav lēkā, tad viņam
piedāvā lēcieni
        if ((myIsJumping == NO_JUMP) && (vdirection < 0)) {
            myIsJumping++;
        } else if (myIsJumping == NO_JUMP) {
            // Ja nelec, tad uzstāda pārvietojumu pa y asi
            RonjaSprite.move(0, 1);
            // Ja personāžs nepieskarās grīdai, tad tam ir jākrīt,
piešķir kritienu
            if (!checkCollision()) {
                myIsJumping = 0;
            }
        }
    }

```

```

        // Pārnes personāžu apakā uz vietu
        RonjaSprite.move(0, -1);
    }
    // ja personāžs lēkā vai krīt
    if ((myIsJumping <= MAX_FREE_FALL) && (myIsJumping !=
NO_JUMP)) {
        myIsJumping++;
    }
    // akcelerē horizontālo kustību
    myIsRunning++;
    // pārbaude uz maksimālo ātrumu
    if (myIsRunning > MAX_SPEED) {
        myIsRunning = MAX_SPEED;
    }
    int horizontal = MOVE_LENGTH * myIsRunning;
    // Nosaka horizontālo pārvietojumu balstoties uz parabolu
    //  $y = (x \cdot x) \cdot (a + b/c)$ 
    // x - lēciena dati, a,b,c - konstantes kas bāzētas uz ekrāna
    izmēra
    int vertical = 0;
    if (myIsJumping != NO_JUMP) {
        vertical = myIsJumping * myIsJumping * JUMP_INT +
(myIsJumping * myIsJumping * JUMP_FRAC_NUM) / JUMP_FRAC_DENOM;
        // piemajai lēciena pusei
        if (myIsJumping < 0) {
            vdirection = -1;
        } else {
            vdirection = 1;
        }
    }
    // nosaka personāža pareizo izskatu balstoties uz
    updateSprite(hdirection, vdirection);
    boolean vcrash = false;
    boolean hcrash = false;
    // nosaka pa pikseļiem, cik tālu personāžs var pārvietoties
    // lēciena laikā katrā no virzieniem:
    while ((vertical >= 1 && !vcrash) || (horizontal >= 1 && !
hcrash)) {
        RonjaSprite.move(0, vdirection);
        if (checkCollision()) {
            RonjaSprite.move(0, -vdirection);
            vcrash = true;
        } else {
            vertical -= 1;
            vcrash = false;
            myViewWindowY += vdirection;
        }
        RonjaSprite.move(MOVE_BUFFER * hdirection, 0);
        if (checkCollision()) {
            RonjaSprite.move(-MOVE_BUFFER * hdirection, 0);
            hcrash = true;
        } else {
            RonjaSprite.move(-MOVE_BUFFER * hdirection, 0);
            RonjaSprite.move(hdirection, 0);
            horizontal -= 1;
            hcrash = false;
            myViewWindowX += hdirection;
        }
    }
    // Ja personāžs ir blokēts vertikāli, tad kritiens apstājas:
    if (vcrash) {

```

```

        myIsJumping = NO_JUMP;
    }
    // Ja personažs ir bloķēts horizontāli, tad nav horizontālas
akseselācijas:
    if (hcrash) {
        myIsRunning = 0;
    }

    if (RonjaSprite.collidesWith(jamesSprite, true)) {
        myCanvas.setGameOver(); //Ja personažs sadurās ar Lielo
Apli, tad spēle beidzās
        return;
    }
    jmove(); // apla pārvietojums
    collapple(); //
    jamesSprite.nextFrame(); //Apla animācija
}

private void collapple() { // pārbaude vai personāžs saskarās ar kādu
//no āboliem, ja tā tad pieskaita punktus un padara šo ābolu neredzamu

    if (RonjaSprite.collidesWith(apple, true)) {
        score++;
        apple.setVisible(false);
        return;
    }
    if (RonjaSprite.collidesWith(a2, true)) {
        score++;
        a2.setVisible(false);
        return;
    }
    if (RonjaSprite.collidesWith(a3, true)) {
        score++;
        a3.setVisible(false);
        return;
    }

    if (RonjaSprite.collidesWith(a1, true)) {
        score*=3;
        a1.setVisible(false);
        return;
    }
    if (RonjaSprite.collidesWith(a4, true)) {
        score++;
        a4.setVisible(false);
        return;
    }
    if (RonjaSprite.collidesWith(a8, true)) {
        score++;
        a8.setVisible(false);
        return;
    }
    if (RonjaSprite.collidesWith(a5, true)) {
        score++;
        a5.setVisible(false);
        return;
    }
    if (RonjaSprite.collidesWith(a6, true)) {
        score++;
        a6.setVisible(false);
        return;
    }

```



```

    }
    if (RonjaSprite.collidesWith(a7, true)) {
        score++;
        a7.setVisible(false);
        return;
    }
}

private void jmove() { //pārvieto Lielo Apli
    mov++; if (mov>50) {mov=-50;}
    if (mov>0) {jamesSprite.move(0, 2);} else {jamesSprite.move(0,
-2);};

}

/**
 * metode kuru izmanto metode requestMove. Nosaka personāža
izskatu, atkarībā no pārvietoējuma
 */
private void updateSprite(int hdirection, int vdirection) {
    // ja personāžs virzās pa labi, vai pa kreisi, to attiecīgi
pārvaido

    if (hdirection > 0) {
        RonjaSprite.setTransform(Sprite.TRANS_NONE);
    } else if (hdirection < 0) {
        RonjaSprite.setTransform(Sprite.TRANS_MIRROR);
    }

    lastHorDirection = hdirection;
    lastVertDirection = vdirection;

}

//-----
// spraitu mijiedarbība
/**
 * Pārbauda vai personāžs pieskaras grīdai
 */
boolean checkCollision() {

    return RonjaSprite.collidesWith(myWalls, true);
}

```

#### Terminu apraksts

API - lietojumprogrammas saskarne

CLDC - Konfigurācija, kas izveidota priekš ierīcēm, kurām ir ierobežoti resursi kā mobilie telefoni tiek sauktā par savienoto ierīču konfigurāciju.

Darbvirsma - Displeja ekrāna apgabals, kas imitē galda virsmu. Darbvirsma dod iespēju lietotājam pārvietot objektus, sākt un beigt uzdevumu izpildi tāpat kā uz reālas galda virsmas, tādējādi atvieglojot lietotāja darbu ar datoru.

Fails - Datu kopa, tekstuāls vai grafisks dokuments, ko glabāšanas, pārsūtīšanas vai apstrādes procesā uzskata un identificē kā vienotu veselumu un kas parasti sastāv no vienādas struktūras ierakstiem.

Implementēšana - Abstrakta shēmas vai algoritma apraksta konkrēts īstenojums.

Spraits - divdimensiju animētais attēls, kas integrēts kopējā ainā

Tiled Layer - vizuālais elements, kas veidots ar šūnu laukumu. Ļauj veidot lielus spēļu laukumus, kuri sastāv no nelieliem objektiem(spraitiem).

Midlets - MIDlets ir JAVA lietojuma struktūra priekš Mobilā informācijas ierīces profila (MIDP), kas parasti implementēta uz Java ME tehnoloģijas atbalstītas ierīces.

Lejupielāde - Datnes pārsūtīšana no attāla datora lietotāja datoram, izmantojot modemu. Plašākā nozīmē — datnes pārsūtīšana no lielāka (centrālā) datora uz tam pakļautu attālu vai mazāku datoru.

Verificēšana - Programmas korektuma formāla pierādīšana

Validēšana - Sistēmas pārbaude, ko veic tās izstrādāšanas beigu posmā, lai pārliecinātos, ka izstrādātās sistēmas funkcionēšana atbilst iepriekš formulētajām prasībām.

Utilītprogramma - Neliela programma vai šādu programmu kopa, kas nodrošina papildpakalpojumus, ko nesniedz oprētājsistēma. Personālajiem datoriem parasti ir daudz atkārtoti izpildāmu uzdevumu (cieto disku defragmentēšana, reti lietojamu datņu saspiešana, datņu atkopšana u. c.), ko veic utilītprogrammas.