Tallinna Ülikool

Informaatika Instituut

XNA Game Studio õppematerjal

Seminaritöö

Autor: Tambet Paljasma

Juhendaja: Jaagup Kippar

Tallinn 2010

Sisukord

Sissejuhatus

Käesoleva seminaritöö eesmärgiks on anda kerge ülevaade Microsoft XNA-st ning koostada õppematerjal (juhend), mille järgi oleks võimalik luua väike mäng.

Õppematerjali koostamisel on silmas peetud eelkõige gümnaasiumis õppivaid programmeerimishuvilisi noori. See sobib kasutamiseks nii informaatikatundides kui ka näiteks IT huvialaringis.

Seminaritöös oleva õppematerjali järgi, mis hõlmab elementaarseid C# osi, on mängu loomine võimalik kõigile programmeerimishuvilistele, olenemata nende varasematest oskustest. Õppematerjal hõlmab uue projekti loomist, objektide tekitamist ekraanile ning nende liigutamist.

Seminaritöö sisaldab ka ülevaadet ja analüüsi tagasisidest, mis laekus isikutelt, kes testisid koostatud õppematerjali selle järgi reaalselt mängu luues.

Antud teema valiku ajendiks oli autori isiklik huvi Microsoft XNA vastu ning väga vähene eestikeelne informatsioon internetis. Ülevaade eestikeelsest informatsioonist, mis on internetis Microsoft XNA kohta võimalik leida, on ära toodud töö 2. peatükis.

1. Microsoft XNA

Microsoft XNA on tööriistade ja vahendite kogum, mille on välja töötanud firma Microsoft, eesmärgiga lihtsustada arvutimängude loomist, arendamist ning haldust. XNA püüab säästa mängu arendajaid liigsest koodikirjutamisest ning liita erinevad mängu loomisel vajalikud aspektid ühte süsteemi. XNA tööriistakogu tehti teatavaks 24. märtsil 2004 San Jose-s, Californias. XNA Game Studio 2.0 nägi päeva valgust detsembris 2007, millele järgnes XNA Game Studio 3.0 oktoobris 2008. XNA Game Studio 4.0 Beta ilmus 12. juunil 2010 koos Windows Phone 7 Development Tools Beta-ga.

(Microsoft Corporation, Microsoft Game Studio [WWW] http://msdn.microsoft.com/en-US/library/bb200104%28v=XNAGameStudio.10%29.aspx(15.10.2010))

1.1 XNA Game Studio 3.0

XNA Game Studio 3.0 , mis on mõeldud Visual Studio 2008 või vabavaralisele Visual C# 2008 Express Edition, lubab luua ka mänge Zune platformile ja toetab ka Xbox Live. Beta versioon Game Studio 3.0-st sai avalikkusele kättesaadavaks septembris 2008. Lõplik versioon ilmus 30. oktoobril 2008. XNA Game Studio 3.0 toetab Visual Studio 2008 versioone ning keeli nagu C# 3.0 ja LINQ.

(Microsoft Corporation,Game Studio 3.0 [WWW] http://msdn.microsoft.com/en-US/library/bb200104%28v=XNAGameStudio.30%29.aspx(14.10.2010))

Käesoleva seminaritöö õppematerjal on mõeldud XNA Game Studio versiooni 3.0 või 3.1 jaoks.

2. Internetist leitav eestikeelne informatsioon XNA Game Studio kohta

Internetis leitav eestikeelne informatsioon XNA Game Studio kohta on väga vähene. Googli otsingumootoris otsingusõna „XNA Game Studio“ annab eestikeelsetelt lehekülgedelt vähe kasulikke vastuseid.

Kasulikuks võib pidada Andres Sireli blogi aadressil http://blogs.msdn.com/b/evangelism/archive/2008/08/25/microsoft-annab-eesti-ppuritele-tasuta-tarkvara.aspx, mis sisaldab informatsiooni DreamSpark-i kohta. DreamSparki annab täiesti tasuta ligipääsu kõigile Eesti tudengitele ning õpilastele Microsofti tehnilisele tarkvarale, millesse kuulab ka Microsoft XNA Game Studio.

XNA Game Studio huvilisel, kes otsib materjali teema kohta, on kõige kasulikum endale tellida „Sams Teach Yourself Microsoft XNA Game Studio Express in 24 Hours“, „Microsoft XNA Game Studio 3.0“ või „Microsoft XNA Game Studio Express“, kuid kahjuks ühtegi neist raamatutest ei ole tõlgitud eesti keelde.

Vähest informatsiooni teema kohta võib leida veel ka hinnavaatluse foorumist aadressil http://foorum.hinnavaatlus.ee/viewtopic.php?t=281256&start=0&postdays=0&postorder=asc&highlight=&sid=7f2cb9bf97873d74fc17cc043eb152ed , kus räägitakse vähesel määral XNA Game Studiost, kuid kompaktset kasutatavat eestikeelset materjali internetis kahjuks ei leidu.

Käesolev töö on keskendunud peamiselt eestikeelse materjali tutvustamisele, kuid kellel on rohkem huvi XNA Game Studio kohta, või külastada inglisekeelseid lehekülgi. Parimaks valikuks oleks http://msdn.microsoft.com/en-us/aa937791.aspx. Sealt võib leida hulgaliselt informatsiooni ja kasulike õpetusi.

3. Õppematerjal

Käesoleva XNA Game Studio õppematerjali koostamisel on kasutatud palju täpsustavaid fotosid ning üksikasjalikke seletusi, kuna on arvestatud asjaoluga, et õppematerjali kasutajatel ei pruugi olla erilisi varasemaid teadmisi käsitletava teema osas. Peamise sihtrühmana on silmas peetud programmeerimishuvilisi gümnaasiumiõpilasi. Loomulikult võivad õppematerjali kasutada ka kõik teised XNA Game Studioga tutvuda soovivad isikud. Õppematerjalis sisalduvad ekraanifotod on tehtud töö autori poolt materjali koostamisel.

Õppematerjali lihtsustamiseks on kõik selles sisalduvad koodiread värvilised. Tuleb rangelt jälgida, et ka kasutaja poolt kirjutatud koodiread muutuksid sama värvi, mis juhendiski. Kui see nii ei ole, siis on see märk tekkinud veast koodi kirjutamisel.

3.1 Uue projekti loomine

XNA rakenduse loomiseks peab esmalt olema arvutisse installitud Microsoft Visual C# Express Edition ja seejärel sellele lisaks XNA Game Studio 3.1. Mõlemad võib leida aadressilt http://creators.xna.com/en-US/downloads.

Kui programm installitud ja käima pandud, tuleb luua uus projekt, kuhu hakata enda rakendust looma.

Uue projekti loomiseks tuleb ülevalt vasakust nurgast valida File ning seejärel New Project.

Foto 1. Uus projekt

Tekkinud aknast saame määrata soovitava projekti nime ja tüübi. Vasakul olevas lahtris saame valida, millist koodiraamistikku kasutame. Koodiraamistik loob kasutajale "raami", mille järgi on kergem jälgida, mida ja kuhu on vaja lisada, et hiljem rakendus luua. XNA Game Studio on mõeldud mängude loomiseks, seega luuakse raamistik, mis seda lihtsustab. Selle mängu loomiseks tuleb vasakult valida XNA Game Studio 3.1. Kuna tehtav mäng on mõeldud arvutile, tuleb selekteerida Windows Game (3.1). Edasi oleks soovitatav panna enda projektile nimi ning valida koht, kuhu see uus projekt luuakse. Nimi võib olla ükskõik milline, kuid soovitatavalt ei tohiks see sisaldada täpitähti ega tühikuid. Kui nimi pandud ja koht projektile valitud, võime vajutada OK, mille järel luuake meile uus tühi projekt, kuhu saame hakata enda rakendust looma.

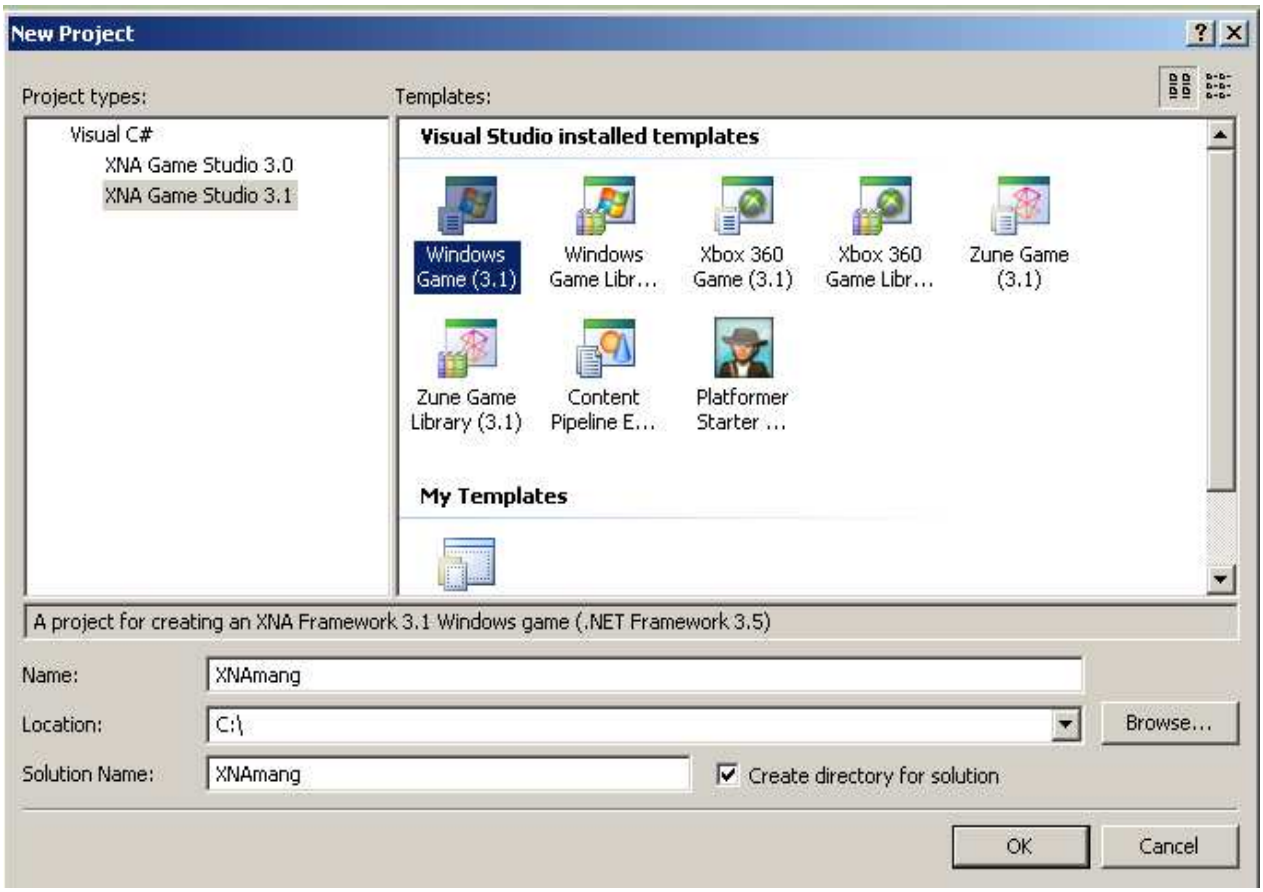

Foto 2. Windows Game

Automaatselt avatakse XNA Game Studio poolt loodud Game1.cs fail, milles toimub ka selle mängu peamine koodi kirjutamine. Paremal ääres asub Solution Explorer aken, mille sees näeme kõiki selles projektis olevaid faile ja katalooge. Sealt leiab ka ekraanile automaatselt avatud Game1.cs faili. Kõige üleval on näha ka Solution-it, millele järgneb ülakomade vahel sinu loodud projekti nimi ning number mitut projekti see solution sisaldab. Iga projekt on

mingi solution-i osa, kusjuures üks solution võib sisaldada mitut erinevat projekti, kuid see mäng piirdub hetkel kõigest ühega. Solution sisaldab peale Game1.cs faili ka teisi faile, mida selle rakenduse loomise juures muutma ei pea.

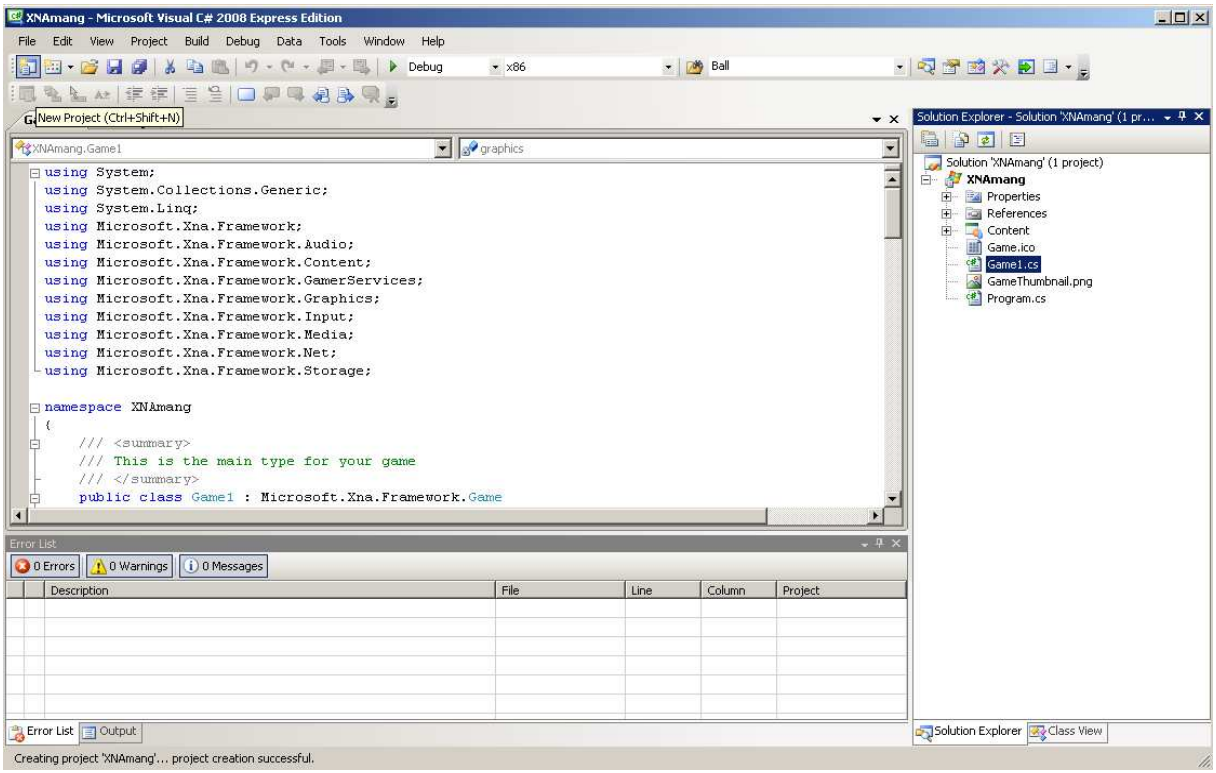

Foto 3. Solution

Nüüd, kus peaks olemas olema üldine ülevaade tekitatud projektist, võib alustada mängu loomisega. Alustuseks tuleb enda projektile lisada objekt, mida hiljem saab vastavalt koodile ekraanile kuvada, liigutada või kontrollida. Solution Explorer kasutab sarnast kataloogi struktuuri nagu Windows. Võimalik on lisada, eemaldada ja liigutada katalooge või faile ning muuta nende nimesid. Selle mängu valmistamiseks tuleb lisada enda projektile pildi fail, mis sisaldab objekti, mille võib igaüks ise teha või kasutada õpetusega kaasas olevat faili pall.png.

Isetehtud pildi juures on soovitatav kasutada laiendit .png. Kui pilt on loodud või allalaetud, tuleb tagasi minna oma projekti juurde. Solution Exploreris tuleb teha parema hiireklahviga klõps Content kataloogil ning seejärel valida *Add*, mis tähendab "lisama" ning *Existing Item,* mis tähendab "olemas olev objekt". Selline tegevus lubab projekti Content kataloogi lisada juba olemasoleva faili. Nüüd tuleb üles leida enda pildi fail, sellele klikkida ning vajutada nupule *Add*. Lisatud pilt kopeeritakse Content kataloogi, seega ei pea muretsema, kui algse failiga peaks midagi juhtuma.

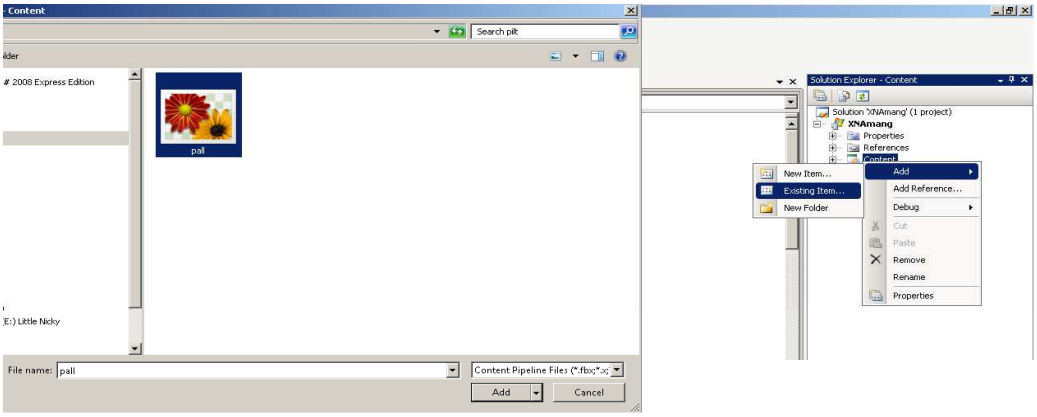

Foto 4. Pildi lisamine

3.2 Uute objektide/muutujate loomine

Nüüd, kus projekt on loodud ning sinna lisatud ka sobiv pilt, võib alustada koodi kirjutamisega.

Kõigepealt tuleb veenduda, et avatud oleks Game1.cs fail. Kui mingil põhjusel ei ole fail teie peaaknas, siis saab selle uuesti Solution Explorer-ist avada. Kui fail avatud võib märgata, et XNA Game Studio on kasutajate eest palju tööd ära teinud ja kõik vajalik rakenduse käivitamiseks, töölhoidmiseks, uuendamiseks ja sulgemiseks on juba olemas. Kui üleval tööriistaribalt vajutada rohelise noolega nuppu (*Start Debugging*) või kasutada klaviatuuril nuppu F5, peaks ekraanile tekkima uus aken sinise taustaga. Meie peame lihtsalt hoolitsema selle eest, et sinna aknasse tekiks meie soovitud pilt.

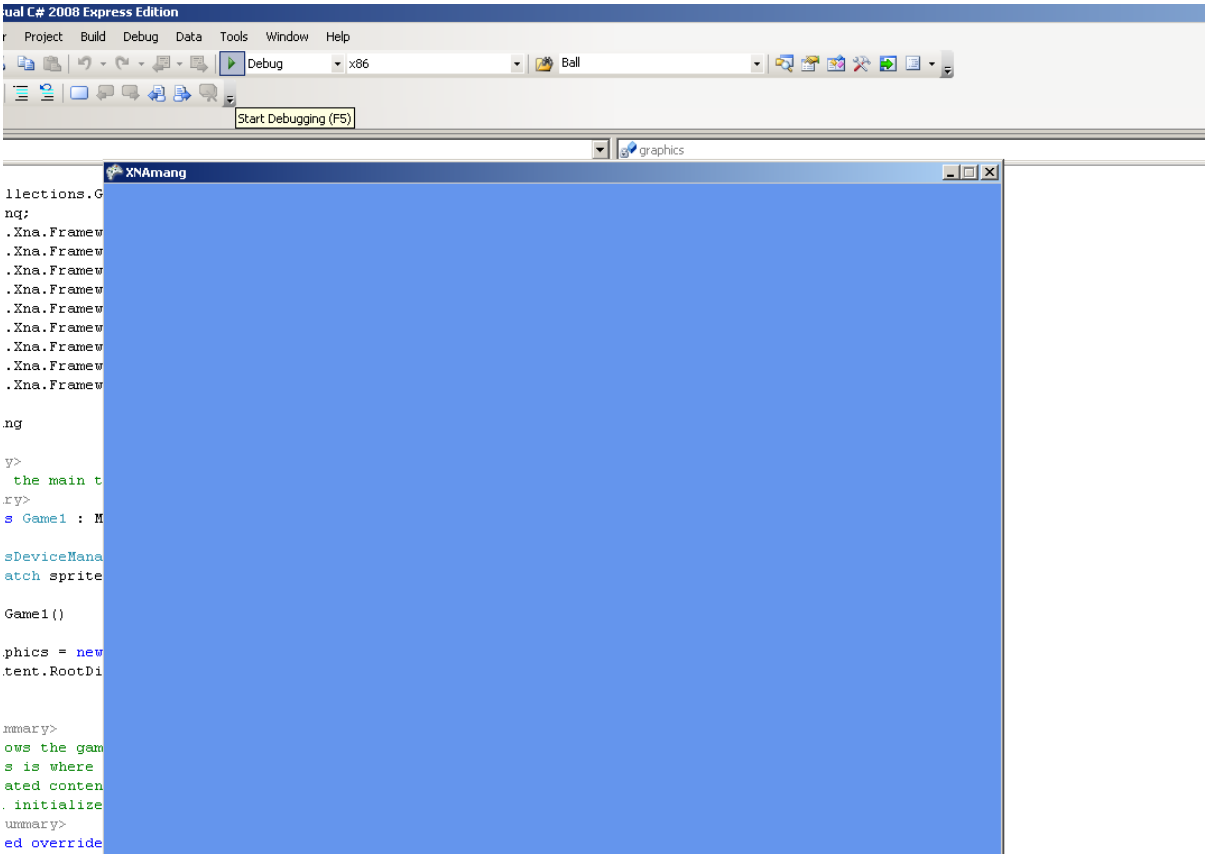

Foto 5. Debug

Game1.cs falis ülevalt alustades näeme ridu, mis algavad sõnaga *using*. Neid käske on vaja, et XNA Game Studio oskaks koodist aru saada ja seda õigesti kasutada.

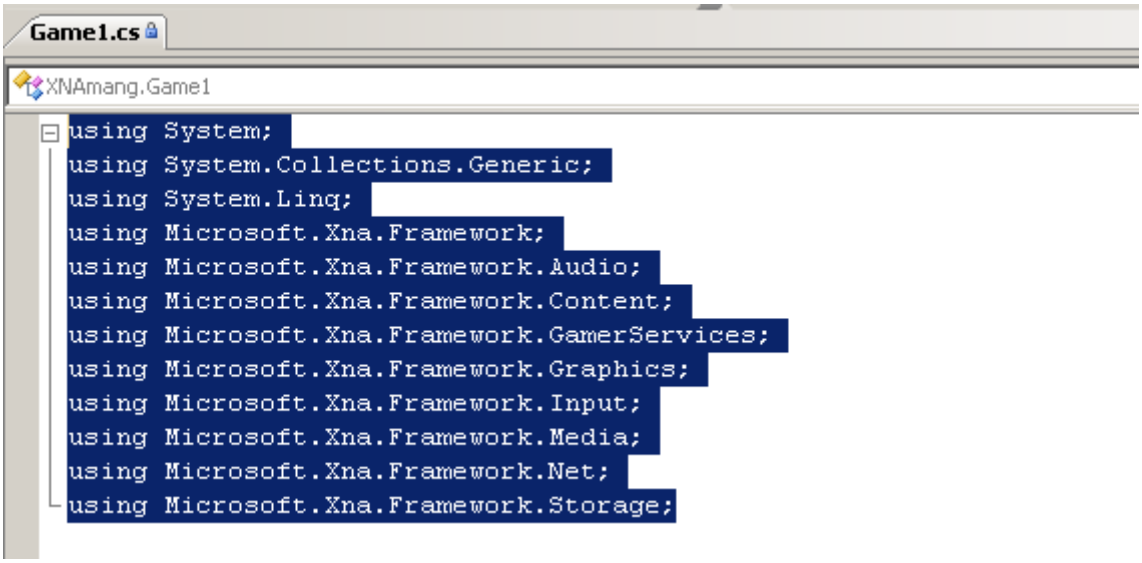

Foto 6. Using

Edasi on näha sinises kirjas *namespace*, millele järgneb loodud projekti nimi. All olev loogeline sulg näitab, et kõik kuni lõppeva loogelise suluni on selle *namespace* sees. Sulule klikkides muutuvad algav ja lõppev sulg aktiivseks (värvitakse halliks).
Namespace'i kasutatakse objektide, klasside, funktsioonide ja meetodite grupeerimiseks.

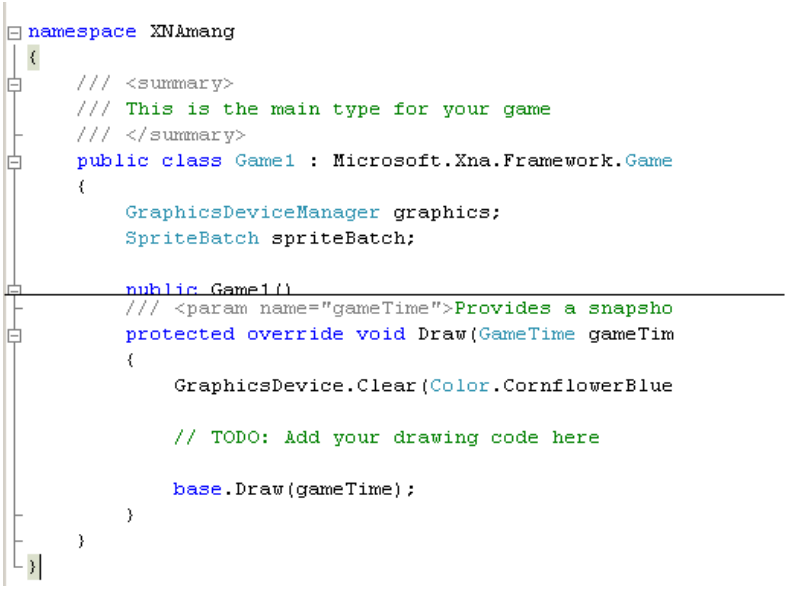

Foto 7. Namespace

Järgmiseks on näha klassi nimega *Game1*. Klass sisaldab andmeid ja juhiseid, mille järgi ta töötab. Erinevad klassid saavad omavahel suhelda ja andmeid vahetada. *Game1* on selle mängu peamine klass, mis hoolitseb rakenduse käima panemise, töötamise, joonistamise ja sulgemise eest. Kogu esimese punkti töö toimub *Game1* klassi sees.

Game1 klassi kirjutame ka oma esimesed koodiread:

```
public class Game1 : Microsoft.Xna.Framework.Game
{
    GraphicsDeviceManager graphics;
    SpriteBatch spriteBatch;
    Vector2 palliAsukoht = new Vector2(0, 0);
    Texture2D palliTekstuur;
```

Vector2 on XNA raamistiku objekt, seda kasutatakse, et salvestada 2D koha informatsiooni. Koht määratakse X ja Y koordinaatide järgi.
Texture2D on teine objekt, mida kasutatakse pildi ekraanil hoidmiseks.

3.3 Pildi üleslaadimine XNA mängu

Järgmiseks tuleb oma pilt mängu laadida. Seda saab teha mõne koodirea lisamisega *LoadContent* meetodisse. *LoadContent´i* ei pea looma, see on juba Game1.cs faili üks osa. Lisaks on paljudes kohtades rohelises kirjas kommentaare. Kommentaarid sisaldavad informatsiooni, mis aitab kasutajal mängu luua.

```
protected override void LoadContent()
{
      // Create a new SpriteBatch, which can be used to draw
      textures.
            spriteBatch = new SpriteBatch(GraphicsDevice);

      // TODO: use this.Content to load your game content here
      palliTekstuur = this.Content.Load<Texture2D>("pall");
}
```

Faili lisatud rida kasutame pildi "pall" laadimiseks. Selleks loome puntki, mille programm kujutab *Texture2D* objektina.

3.4 Pildi kuvamine ekraanile

Nüüd kus *Texture2D* objekt on üleslaetud, on võimalik ta ekraanile kuvada. Kõik mida ekraanil näidatakse käib läbi *Draw* meetodi , mis on Game1.cs failis juba olemas. *Draw* meetod uuendab pidevalt objektide joonistamist ekraanile, tagades nende ekraanil püsimise ja tehtavad muutused. *Draw* meetodisse tuleb lisada järgnev kood:

```
protected override void Draw(GameTime gameTime)
{

graphics.GraphicsDevice.Clear(Color.CornflowerBlue);

   // TODO: Add your drawing code here

   spriteBatch.Begin();
   spriteBatch.Draw(palliTekstuur, palliAsukoht, Color.White);
   spriteBatch.End();

   base.Draw(gameTime);

}
```

SpriteBatch objekt joonistatakse ekraanile automaatselt, kui mänguaken avatakse. *SpriteBatch* objekte kasutatakse, et joonistada 2D pilte ekraanile. Nende kasutamiseks tuleb kõigepealt alustada joonistamist käsuga *Begin* .

Kui see on tehtud, siis on võimalik joonistada ekraanile kõik pildid, mida kasutaja tahab. See mäng sisaldab hetkel ainult ühte pilti, kuid kogukamad mängud võivad sisalda sadu pilte ja *Draw* käsklusi. *Draw* käsklusega saab määrata tekitatava pildi tekstuuri, positsiooni ning pildil oleva varju värvi. Pannes värviks *White*, ei lisata pildile varju.
Kui kõik vajalik on lisatud, saab käsuga *End* kõik tehtu ekraanile joonistada. Lõpetuseks tuleb vajutada *Star Debugging* nuppu (või vajutada klaviatuuril F5 klahvi) ning pall peaks tekkima uue akna üles vasakusse nurka.

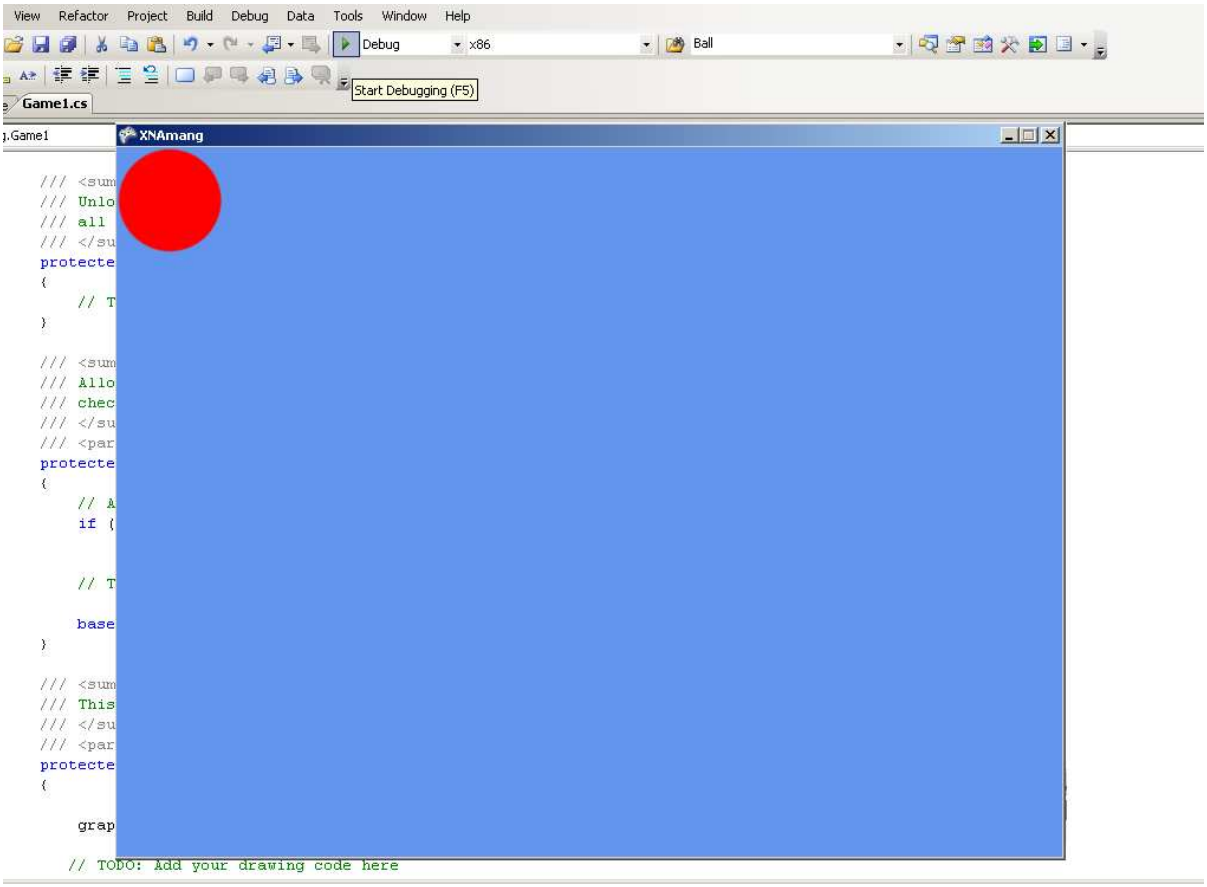

Foto 8. Pall

3.5 Mitme palli kuvamine ekraanile

Nüüd, kus on võimalik pilte ekraanile joonistada, oleks soovitatav luua uus fail, et lihtsustada mitme pildi joonistamist ekraanile.

Uue faili loomiseks tuleb hiire parema klahviga vajutada oma projektile, seejärel valida *Add* ning *New Item*. Tekkinud aknast peab üles otsima ikooni *Class*. Kui ikoon on leitud, tuleb see selekteerida ning oma uuele failile nime anda, näiteks Pall.cs .

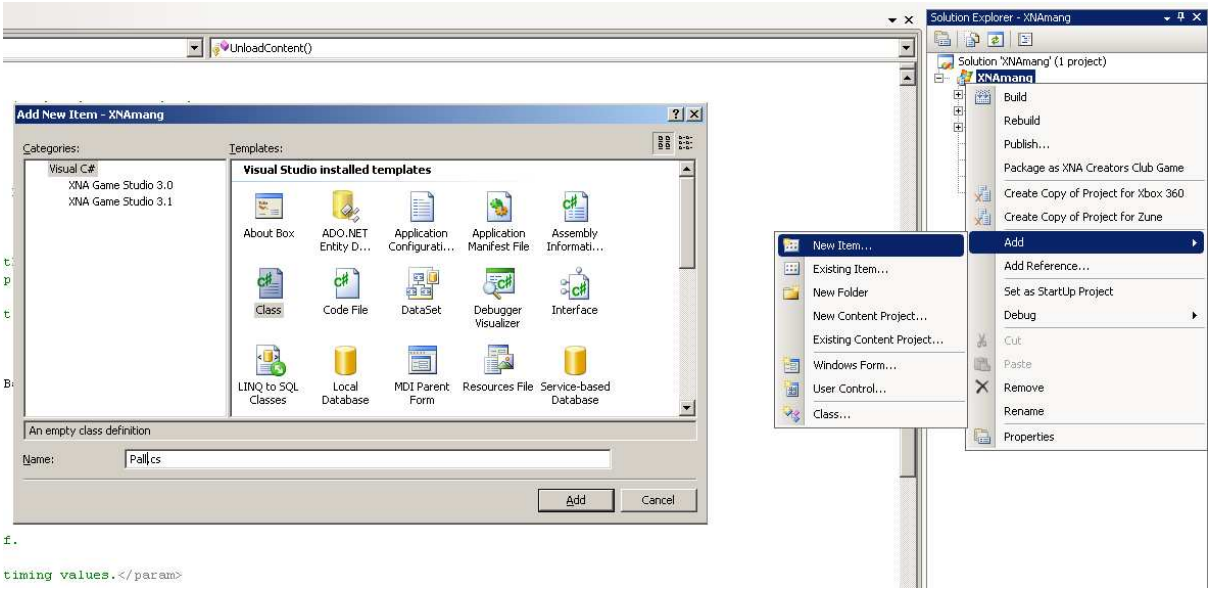

Foto 9. Add content

Nüüd peaks uus klassi fail ekraanile tekkima ja ka Solution Exploreri-sse. Nagu näha sisaldab uus fail palju vähem *using* algusega ridu, mis tähendab, et peame need Game1.cs failist ka siia kopeerima.

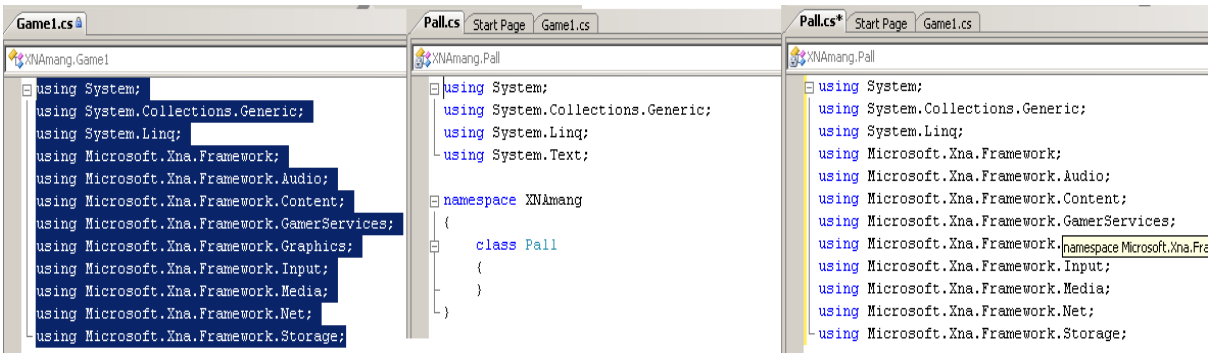

Foto 10. Kopeerimine

Edasi tuleb lisada uude faili paar rida koodi, mis hoolitseb meie palli tekstuuri ja positsiooni eest.

```
class Pall

    {

        public Vector2 Asukoht = new Vector2(0, 0);

        private Texture2D Tekstuur;

    }
```

Nagu näha on kood eelnevast natuke erinev. *Asukoht* objekti ette on lisatud *public* ja *Tekstuur*`i ette *private*. See tähedab, et objekti *Asukoht* saab muuta ka väljaspoolt seda faili, objekti *Tekstuur´*i mitte.

Järgmiseks tuleb luua uus *LoadContent* meetod. Meeles tuleb pidada seda, et *LoadContent* meetod peab olema *class Pall* loogeliste sulgude vahele, näitamaks, et see meetod käib klassi *Pall* alla. See meetod teeb täpselt sama *Pall* klassile, mida *LoadContent* teeb *Game1* klassile. Uue meetodi loomiseks tuleb kirjutada:

```
public void LoadContent(ContentManager theContentManager,
string Nimi)

{

    Tekstuur = theContentManager.Load<Texture2D>(Nimi);

}
```

Pall klassi jaoks loodud uuel *LoadContent* meetodil on sulgudes kirjas parameetrid, mida meetod kasutab. Pall klassil ei ole enda *ContentManager*-i seega tuleb see luua. *ContentManager* tahab teada ka pildi nime *(Nimi)* , mis hiljem *Texture2D* objekti laetakse, et seda saaks ekraanil kuvada.

Nüüd, kui pilt on laetud, tuleb see ka ekraanile joonistada. Selleks tuleb luua uus *Draw* meetod enda *Pall* klassi jaoks.

```
 public void Draw(SpriteBatch theSpriteBatch)

 {

     theSpriteBatch.Draw(Tekstuur, Asukoht, Color.White);

}
```

Uus *Draw* meetod teeb *Pall* klassile sama, mida vana *Draw* meetod teeb *Game1* klassile. Ka see meetod tuleb lisada *class Pall* sisse.

Kui uued klassid loodud, tuleb hakata vana Game1.cs failis olevat koodi muutma, nii et pildi tekstuur ja asukoht laetakse *Pall* klassist. Selleks tuleb esiteks kustutada *Game1* klassist asukoha ja tekstuuri objektid ning asendada need Pall klassi omadega. Tuleb lisada ka üks uus rida koodi. Selle koodireaga saab kasutada uut *Pall* klassi.

```
public class Game1 : Microsoft.Xna.Framework.Game
{
    GraphicsDeviceManager graphics;
    SpriteBatch spriteBatch;
    Pall esimene;
```

Järgmiseks tuleb luua pall, nii et seda saaks *Game1* klassis kasutada. Selleks tuleb lisada rida koodi *Initialize* meetodisse, mis on Game1.cs failis juba olemas.

```
protected override void Initialize()
{
        // TODO: Add your initialization logic here
        esimene = new Pall();

        base.Initialize();
}
```

Kui pall loodud, tuleb see ka rakendusse laadida. Selleks peab asendama vana koodirea *LoadContent* meetodis uuega.

```
protected override void LoadContent()

{

// Create a new SpriteBatch, which can be used to draw
textures.

       spriteBatch = new SpriteBatch(GraphicsDevice);


// TODO: use this.Content to load your game content here

        esimene.LoadContent(this.Content, "pall");

        }
```

Nii tehes laetakse objekt *Pall* klassist. Nüüd tuleb veel muuta *Draw* meetodit Game1.cs failist nii, et ekraanile joonistatakse *Pall* klassi objekt.

```
protected override void Draw(GameTime gameTime)
  {

graphics.GraphicsDevice.Clear(Color.CornflowerBlue);


      // TODO: Add your drawing code here

      spriteBatch.Begin();

      esimene.Draw(this.spriteBatch);

      spriteBatch.End();


      base.Draw(gameTime);

 }
```

Kui nüüd *Start Debugging* nuppu vajutada, peaks pall tekkima ekraanile uude kohta.

Nüüd, kus on olemas *Pall* klass, on edaspidi palju lihtsam uusi palle ekraanile tekitada. On ka näha, et pildi joonistamise kood on palju lühem, kuna kõik vajalik informatsioon laetakse uuest *Pall* klassist. Edaspidi on palju lihtsam luua rohkem palle ekraanile, kasutades uut *Pall* klassist laetavat objekti näiteks teine (*Pall teine;*) .

3.6 Palli asukoha muutmine

Selleks tuleb lisada kaks rida koodi *Game1* klassis olevale *LoadContent* meetodile.
Kuna hiljem õpetuses pannakse ka pall liikuma, on mõistlik kasutada x ja y koordinaate eraldi. Selleks tuleb lisada kaks rida uut koodi.

```
esimene.Asukoht.X = 200;

esimene.Asukoht.Y = 300;
```

Ülesanne.

Proovida luua veel vähemalt üks pall ekraanile.

Lahendus.

- Kõigepealt tuleb lisada *Game1* klassi rida `Pall teine;`
- *Initialize* meetodisse rida `teine = new Pall();`
- *LoadContent* meetodisse read `teine.LoadContent(this.Content, "pall");`

  ```
  teine.Asukoht.X = 400;

  teine.Asukoht.Y = 400;
  ```

- Lõpetuseks *Draw* meetodisse rida `teine.Draw(this.spriteBatch);`
- Nüüd *Start Debugging* nuppu vajutades peaks ekraanile ilmuma kaks erikohtades asuvat palli.

3.7 Palli liigutamine ekraanil nuppude abil

Nüüd kus X ja Y on eraldi Game1.cs failis deklareeritud, saab hakata palli liigutama. Selleks tuleb lisada koodi *Update* meetodisse, mis on Game1.cs failis juba olemas. *Update* meetodit kasutatakse ekraanil olevate elementide muutmiseks nagu näiteks millegi liigutamiseks. *Update* meetodit uuendatakse samal kiirusel või isegi kiiremini kui *Draw* meetodit ning kõik tehtud muutused joonistatakse ekraanile kasutades *Draw* meetodit. *Update* meetodisse tuleb sisestada järgnevad read:

```
KeyboardState nupp = Keyboard.GetState();
```

KeyboardState nupp = Keyboard.GetState(); see koodi osa kontrollib, mis seisundis on hetkel arvuti klaviatuur nagu näiteks, kas mõnda nuppu on vajutatud või hoitakse all.

Kohe järgmisele reale tuleb kirjutada:

```
if (nupp.IsKeyDown(Keys.Left)){ pall.Asukoht.X -= 2; }
```

If lauset kasutatakse mingi sündmuse kontrollimiseks ning sellele reageerimiseks.
Siin koodis tähendab see seda, et kui nupp, mis on hetkeline klaviatuuri seisund, tuvastab, et teatud klahvi hoitakse all või vajutatakse ning selleks klahviks on vasak nool*(Keys.Left)*, antakse loogeliste sulgude sees olev käsklus. Siin juhul on selleks käskluseks liigutada pilti mööda X telge 2 ühikut vasakule(*X -= 2*, kus -= on samaväärne eelnevast vastusest 2 lahutamisega).

Ülesanne.

Panna pall liikuma ka üles, alla ja paremale antud käsu korral.

Lahendus.

Panna pall liikuma ka teiste nuppude järgi on päris lihtne. Tuleb ainult muute klahvi nimetust ning, mis telge mööda ja mitu ühikut pall liigub.

Üles:

```
if (nupp.IsKeyDown(Keys.Up)){  pall.Asukoht.Y -= 2; }
```

Alla:

```
if (nupp.IsKeyDown(Keys.Down)){ pall.Asukoht.Y += 2; }
```

Paremale:

```
if (nupp.IsKeyDown(Keys.Right)){ pall.Asukoht.X += 2; }
```

4. Tagasiside analüüs õppematerjali testinutelt

Kuigi XNA Game Studio õppematerjali koostamisel on silmas peetud eelkõige programmeerimishuvilisi gümnaasiumiõpilasi, palusin selle testimisel osaleda siiski väga erinevas vanuses ja kogemustega inimestel. Materjali kasutasid testimise eesmärgil kuus inimest, kellest kaks olid keskealised, kaks üliõpilased ja kaks gümnaasiumiõpilased. Ka arvutikasutamise oskused olid erinevaid - oli nii tavakasutajaid kui ka neid, kes varem programmeerimisega pisut kokku puutunud.

4.1 Õppematerjali testimine

Testijate ülesandeks oli õppematerjali alusel mängu loomine, selgitamaks välja, kas koostatud materjal on piisavalt arusaadav ja selle juhiseid järgides mängu loomine võimalik või mitte. Kasutajatele esitati seitse küsimust, millele nad pidid peale õppematerjali läbitöötamist vastama.

Alljärgnevalt on ära toodud esitatud küsimused ja iga küsimuse järel lühikokkuvõte saadud vastustest.

1. Milline õppematerjali alateemadest tundus sulle kõige raskem?

Enamus õppematerjali läbinutest leidsid, et kõige raskem alateema oli „Mitme palli kuvamine ekraanile“. Raskeks peeti ka veel alateemat „Uue faili loomine“.

2. Milline alateema oleks võinud olla paremini selgitatud?

Kuna kõige raskemat alateemaks peeti „Mitme palli kuvamine ekraanile“, siis oleks võinud see teema olla ka paremini seletatud.

3. Kas sul tekkis õppematerjali läbitöötades probleeme?

Kahel õppematerjali kasutanutest tekkis probleeme, kuid need oli tingitud kasutaja tehtud väikesest koodiveast või õppematerjali mitte järgimisest.

4. Kas sul tekkis peale õppematerjali läbimist küsimusi?

Ühelgi õppematerjali kasutanutest ei tekkinud peale selle läbimist lisaküsimusi.

5. Kas õppematerjali abil tehtav oli sulle arusaadav?

Õppematerjali läbinud leidsid, et kõik õppematerjali abil tehtu oli arusaadav, kuid mõni läbinutest lisa ka, et ilma Microsoft Visual C# programmita oleks materjali läbimine palju raskemaks osutunud.

6. Mis sulle selle õppematerjali juures meeldis?

Kasutajate sõnul oli kõige meeldivam õppematerjali juures see, et kõik oli ette antud ja koos selgituste täpse jälgimisega oli materjali läbimine lihtne ning arusaadav.

7. Kas näed võimalust õpitut ka hiljem kasutada?

Õpitu hilisema kasutamise võimalust nägid just gümnaasiumiõpilased, ka üks üliõpilastest. Keskealised testijad arvasid, et nad seda hiljem enam ei kasuta.

4.2 Analüüsi kokkuvõte

Õppematerjali alusel mängu loonute meelest oli materjal meeldivalt kerge ja selle alusel mängu loomine täiesti võimalik. Õpetusi täpselt järgides oli kõik tehtav ning arusaadav. Leidus ka väiksemaid vigu ja kõik ei toiminud kohe nii, nagu vaja, kuid juhendit uuesti ning täpsemalt lugedes saadi kõik vead parandatud. Need õppematerjali testijad, kellel ei olnud mingit varasemat kokkupuudet programmeerimisega, said lisaefektina veel ka esmase ettekujutuse, mis õieti on programmeerimine ja kuidas see töö käib. Üldiselt läks materjali läbitöötamine sujuvalt ning kõik kasutanud jäid sellega rahule.

Enim huvi pakkus käsitletud teema just gümnaasiumiõpilastele, sest nad nägid võimalust oma IT-alaste teadmiste laiendamiseks ja usuvad, et saavad ka hiljem omandatud oskusi kasutada. Materjali läbitöötamine andis neile julgust ka edaspidi proovida tegeleda programmeerimisega.

Seminaritöö kokkuvõte

Käesoleva seminaritöö tulemusena on koostatud XNA Game Studio õppematerjal programmeerimishuvilistele noortele, kes alles valmistuvad tegelema programmeerimisega ja ei oma veel erilisi kogemusi selles vallas. Materjali alusel on võimalik luua väike “mäng” .

Sihtrühmaks, kellele õppematerjal on suunatud, valiti süvendatud infotehnoloogia huviga gümnaasiumiõpilased, aga positiivset tagasisidet sai materjal peale nende ka teistelt vanuserühmadelt. Eesmärk oli luua õppematerjal piisavalt lihtne ja kergesti käsitletav, et see ei peletaks kasutajaid oma keerukusega eemale, vaid tekitaks huvi ja lõpptulemus oleks saavutatav paljudele. Tagasisidet arvestades tundub, et see õnnestus.

Õppematerjalis kirjeldatud teemat gümnaasiumi IT tundides reeglina ei käsitleta. See sobib aga hästi näiteks IT huvialaringis kasutamiseks. Õppematerjal annab õpilastele võimaluse hakata tegema esimesi samme programmeerimise vallas ja äratab edasist huvi teema vastu.

Materjali alusel loodav mäng on väga algeline, aga see loob ettekujutuse süsteemi toimimisest ja võimaldab õpilasel endal huvi korral jätkata juba keerulisemate mängude programmeerimist.

Töö autorile oli õppematerjali loomise protsess uudne kogemus. Alati ei olnudki lihtne konkreetselt ja üheseltmõistetavalt kõiki tegevusi kirja panna. Samuti oli oluline pidevalt sihtrühma silmas pidada, et mitte minna liiga keeruliseks. Ära tuli seletada ka mõni väike detail, mis endale tundus nii iseenesestmõistetav, kuid õppurile ei pruugi seda olla.

Kasutatud allikad

Microsoft Corporation, XNA Creators Club Online [WWW] http://creators.xna.com/en-US/ (18.09.2010)

Microsoft Corporation, XNA Developer Center[WWW]http://msdn.microsoft.com/en-gb/xna/default.aspx (25.10.2010)

Andres Sirel, Microsoft annab Eesti õppuritele tasuta tarkvara[WWW] http://blogs.msdn.com/b/evangelism/archive/2008/08/25/microsoft-annab-eesti-ppuritele-tasuta-tarkvara.aspx (20.10.2010)

Bond666, Microsoft XNA GameStudio Express (Beta) [WWW] http://foorum.hinnavaatlus.ee/viewtopic.php?t=281256&start=0&postdays=0&postorder=asc&highlight=&sid=7f2cb9bf97873d74fc17cc043eb152ed (20.10.2010)