8 DARBS AR LIETOJUMPROGRAMMU INSTALĀTORIEM

Jebkuras nopietnas lietojumprogrammas izstrāde ietver arī tās artefaktu sagatavošanu instalēšanai gala lietotāju datoros. Līdz ar to šajā laboratorijas darbā apskatīsim izstrādes vides piedāvātās iespējas un vizuālos rīkus instalātoru sagatavošanai.

Šajā laboratorijas darbā lietotie termini:

- Instalēšana – interaktīvs vai automatizēts process, kurā lietojumprogrammas artefakti tiek uzstādīti un sagatavoti izpildīšanai gala lietotāja datorā;
- Atinstalēšana – process, kurā lietojumprogrammas artefakti tiek izdzēsti no gala lietotāja datora, un kura beigās lietojumprogramma vairs nav pieejama;
- Instalācija – gala lietotāja datorā uzstādītas lietojumprogrammas artefaktu un uzstādījumu kopums;
- Instalātors jeb instalēšanas programma – speciāla lietojumprogramma, ar kuras izpildīšanu veic lietojumprogrammas instalēšanas un atinstalēšanas procesus, bieži vien arī pārinstalēšanu un jaunināšanu;
- Instalātora projekts – izstrādes vides projekts, kura gala artefakts ir lietojumprogrammas instalātors.

8.1 INSTALĀTORU VEIDOŠANA

Kaut arī varētu šķist, ja lietojumprogramma sastāv tikai no pāris artefaktiem (piemēram, izpildāmā faila ar paplašinājumu .EXE un dinamiski ielādējamām tipu bibliotēkām ar paplašinājumu .DLL), tad vajadzība pēc instalātora ir lieka. Tomēr pie praktiski jebkuras lietojumprogrammas uzstādīšanas gala lietotāju datorā, ir jāveic noteiktas darbības, kuru izpildi var uzticēt specializētām lietojumprogrammām, kuras sauc par instalātoriem. Piemēram, šādi instalātori, kā minimums varētu veikt:

- lietojumprogrammas instalācijas kataloga izvēli, atkarībā gala lietotāja datora operētājsistēmas versijas vai lietotāja vēlmēm;
- saīsnes izveidošanu lietojumprogrammas piekļuvei no programmu izvēlnes vai ekrāna darbvirsmas;
- datu avota saites izveidošanu; u.c.

Ņemot vērā, ka lietojumprogrammas iedalās vairākos tipos, piemēram, komandrindas, grafiskās saskarnes un tīmekļa lietojumprogrammās, tad katram lietojumprogrammu tipam ir specifiskas darbības, kas jāveic to instalēšanai vai nu gala lietotāja datorā vai arī tīmekļa serverī.

Šim nolūkam izstrādes vidē ir pieejamas vairākas instalātoru projektu sagataves, kurās iespējams norādīt lietojumprogrammas artefaktus, kas jāiekļauj instalātora projektā, bet šāda projekta nokompilēšanas gala rezultātā tiek iegūts instalātors, kuru pēc tam var izmantot, lai lietojumprogrammu uzstādītu gala lietotāju datoros.

Instalātors parasti ir fails ar paplašinājumu .MSI (Windows Installer), un sevī iekļauj gan pašas lietojumprogrammas palaišanai nepieciešamos artefaktus, gan arī šo artefaktu izpildei nepieciešamos papildus failus, kā arī citu informāciju par lietojumprogrammu, piemēram, nosaukumu, autoru, licences nosacījumus, instalēšanas instrukcijas. Galvenais, ka instalātors nodrošina, ka vai nu lietojumprogramma uzstādās pilnībā veiksmīgi, vai arī, kļūdu gadījumā, gala lietotāja operētājsistēmas konfigurācija tiek atgriezta tādā stāvoklī, kādā tā bija pirms instalātora palaišanas.

Izstrādes vide piedāvā vēl vienu instalātora projekta veidu – ClickOnce. Tas ir paredzēts mazliet specifiskākiem mērķiem nekā Windows Installer, jo galvenokārt tiek pielietots instalātoru publicēšanai noteiktā vietnē (tīmekļa servera katalogā vai koplietojamā tīkla katalogā), no kurienes

gala lietotāji aktivizē instalātora palaišanu – visas izmaiņas, kas vēlāk tiek veiktas ClickOnce instalātora publicēšanas vietnē, pie uzinstalētās lietojumprogrammas atkārtotas palaišanas, tiek konstatētas un automātiski uzstādītas gala lietotāja datorā. Vēl viena šī instalātora priekšrocība ir tā, ka lietotājam nav jābūt administratora tiesībām, lai veiktu šī tipa instalātoru palaišanu - visa informācija tiek glabāta lietotāja konta apakškatalogā.

Padoms: *Iepriekšējo divu instalātoru funkcionalitāte salīdzinājumu var apskatīt, piemēram, URL adresē http://msdn.microsoft.com/en-us/library/e2444w33.aspx.*

Jaunu instalātoru projektu izveidošanu uzsākt tāpat kā jebkuru citu izstrādes vides projektu – projekta sagataves izvēles dialogā, kategorijas **Other Project Types** apakškategorijā **Setup and Deployment**, izvēloties vienu no iespējamajām instalātoru projektu sagatavēm:

- **Setup Project** – izveido instalātora projektu grafiskās vai konsoles tipa lietojumprogrammām. Instalātorā ievietotie artefakti parasti tiek kopēti gala lietotāja datora „Program Files" apakškatalogā.
- **Web Setup Project** – izveido instalātora projektu tīmekļa lietojumprogrammai, kas ļauj automatizēt tīmekļa lietojumprogrammu izvietošanu tīmekļa servera Internet Information Server virtuālajā katalogā.
- **Merge Module Project** – izveido t.s. instalātora moduļa projektu, kura gala rezultāts (ar paplašinājumu .MSM) nav patstāvīgs instalātors, bet gan to kā nedalāmu vienību var iekļaut iepriekšējo divu instalātoru projektu veidos, piemēram, lietojumprogrammas atsevišķai apakšsistēmai, kas tiek kopīgi izmantota vairākiem **Setup Project** un **Web Setup Project** sagatavju projektiem.
- **CAB Project** – izveido .CAB tipa instalātoru, kas parasti tiek izmantots, ja instalātoru plānots piedāvāt lejupielādēt, izmantojot tīmekļa pārlūku. Šī tipa instalātors tiek izmantots specifiskiem mērķiem un nav domāts vispārīgu lietojumprogrammu instalātoru veidošanai.
- **Setup Wizard** – ar grafiska vedņa palīdzību, ļauj automatizēt iepriekšējo četru projektu sagatavju izveidošanu un instalējamo artefaktu pievienošanu.

Instalātoru projektus parasti apvieno vienā risinājumā ar pašas lietojumprogrammas projektu. Tas ļauj instalātora projektā automātiski iekļaut visus lietojumprogrammas gala artefaktus, un citus tās izpildei nepieciešamos failus (piemēram, nepieciešamo .NET versiju, SQL servera versiju un instanci, u.c.). Līdz ar to, instalātors nodrošina, ka lietojumprogrammas izpildei visi nepieciešamie priekšnoteikumi tiek izpildīti. Piemēram, gala lietotāja datorā vispirms tiek piedāvāts uzinstalēt nepieciešamo .NET ietvara versiju, un kamēr tas netiek izdarīts pašu lietojumprogrammu neļauj instalēt.

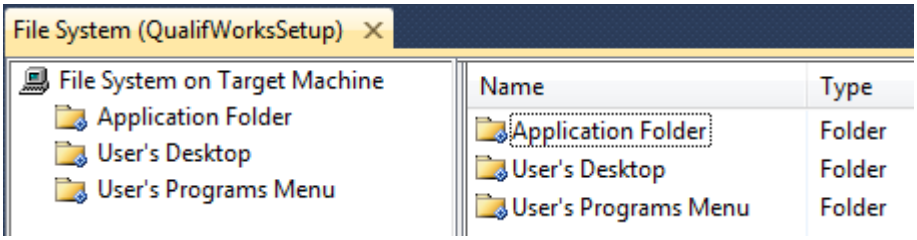
Apskatīsim instalātora projektu iespējas, izveidojot instalātoru noslēguma darbu lietojumprogrammai.

8.2 UZDEVUMS PASNIEDZĒJA VADĪBĀ

Formulēsim šādu uzdevumu: risinājumā **QualificationWorks** izveidot instalātoru projektam **QualifWorksClient**, kas instalēšanas laikā ļauj lietotājam norādīt lietojumprogrammas instalēšanas katalogu, kā arī atbilstoši tam, aktualizē saiti uz datu avota atrašanās vietu. Datu avota sagatavi (fails **QualificationWorks.sdf**) realizēt kā instalātora moduli, kuru iekļaut lietojumprogrammas **QualifWorksClient** instalātora projekta sastāvā.

Šim nolūkam:

- Risinājumā **QualificationWorks**, izveidot jaunu instalātora projektu, izmantojot sagatavi **Setup Project**, kuru nosaukt par **QualifWorksSetup**. Pēc instalātora projekta izveidošanas, automātiski atvērsies tā failu sistēmas redaktora logs:



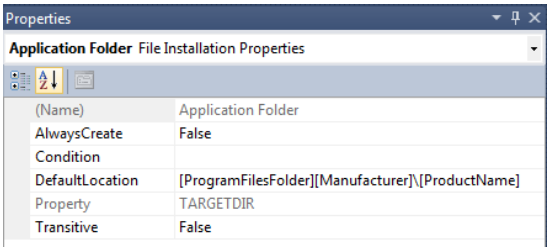
Failu sistēmas redaktors ļauj definēt virtuālus katalogus, kuriem pēc tam piesaistīt noteiktus artefaktus, kas, instalācijas procesā tiks iekopēti gala lietotāja datora fiziskos katalogos. Katalogi tiek saukti par virtuāliem, jo to nosaukumi tikai apraksta loģiskos nosaukumus, nevis nosaukumus gala lietotāja datorā.

Visbiežāk tiek lietoti šādi virtuālie katalogi:

Kataloga nosaukums:	**Nozīme:**
Application Folder	Katalogs, kas apzīmē noklusēto lietojumprogrammas instalēšanas katalogu. Parasti, tas ir speciāli lietojumprogrammai izveidots **"Program Files"** apakškatalogs.
User's Desktop	Lietotāja ekrāna darbvirsmas katalogs, kuru parasti izmanto lietojumprogrammu saīšņu saglabāšanai.
User's Programs Menu	Programmu izvēlnes katalogs. Šajā katalogā tiek ievietotas saīsnes uz lietojumprogrammas izpildāmo failu un dokumentāciju.
Program Files Folder	Instalēto programmu sistēmas katalogs, parasti **"C:\Program Files"**.
System Folder	Operētājsistēmas sistēmas apakškatalogs, kurā tiek glabāti visā sistēmā koplietojami faili - parasti dinamiski ielādējamas bibliotēkas .DLL, kas tiek izmantotas kopīgi vairākās neatkarīgās lietojumprogrammās.
Fonts Folder	Operētājsistēmas fontu katalogs. Izmanto, ja lietojumprogrammas darbībai nepieciešams instalēt specifiskus fontus.
User's Application Data Folder	Katalogs, kurā gala lietotāja palaistajām lietojumprogrammām ir tiesības veidot jaunus failus vai modificēt esošos, kas tiek izmantoti lietojumprogrammas vajadzībām.
Custom Folder	Patvaļīgs virtuālais katalogs, kuru parasti izmanto, ja instalātora vajadzībām neder neviens no standarta virtuālajiem katalogiem. Šim katalogam būtiski ir uzstādīt īpašību **Property** (skat. zemāk).

Minēto un citu virtuālo katalogu pievienošanu veic no failu sistēmas redaktora loga konteksta izvēlnes punkta **Add Special Folder** apakšpunktiem. Ja kādā katalogā nepieciešams veikt vēl detalizētāku grupēšanu, uz atbilstošā kataloga konteksta izvēlnes izvēloties komandu **Add |Folder**, ir iespējams izveidot virtuālā kataloga apakškatalogu, kura nosaukums jau sakritīs ar to, kas tiks izveidots gala lietotāja datorā.

Katram virtuālajam katalogam tiek piešķirts unikāls identifikators, kuru instalēšanas laikā var izmantot, lai iegūtu faktisko šī kataloga ceļu un nosaukumu. Šo identifikatoru definē ar virtuālā kataloga īpašības **Property** vērtību:



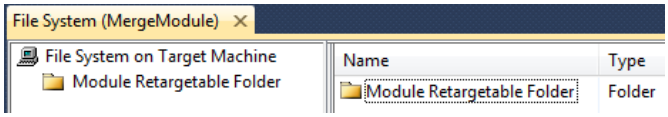
Piemēram, virtuālam katalogam **Application Folder** šis identifikators ir **TARGETDIR**, kurš pēc noklusēšanas satur īpašības **DefaultLocation** vērtību: **[ProgramFilesFolder][Manufacturer]\[ProductName]**. Pēc šīs izteiksmes var secināt, ka lai atsauktos uz standarta vai instalātorā definētiem identifikatoriem, to nosaukumi jāraksta kvadrātiekavās. Identifikatoru **Manufacturer** un **ProductName** vērtības tiek uzstādītas paša instalātora projekta īpašību loga attiecīgajās īpašībās.

Jāuzsver, ka ne visus identifikatorus iespējams izmainīt - piemēram, standarta virtuālo katalogu īpašība **Property** ir atspējota un īpašību logā to nevar mainīt.

Pirms turpināsim ar instalātora projekta **QualifWorksSetup** izstrādi, izveidosim jaunu instalātora moduļa projektu, kura vienīgais uzdevums būs nodrošināt datu avota sagataves faila **QualifWorksData.sdf** uzstādīšanu.

Šim nolūkam:

- Risinājumam **QualificationWorks**, izveidot jaunu instalātora projektu, izmantojot sagatavi **Merge Module Project**, kuru nosaukt par **MergeModule**. Pēc projekta izveidošanas, automātiski atvērsies tā failu sistēmas redaktora logs, kurā pēc noklusēšanas ir tikai viens virtuālais katalogs:

  Virtuālais katalogs **Module Retargetable Folder** ir specifisks tikai instalātoru moduļiem, jo šī kataloga faktisko galamērķi obligāti jāpārdefinē instalātora projektā, kurā tiek iekļauts instalātora modulis.

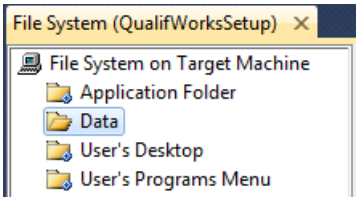
- Virtuālā kataloga **Module Retargetable Folder** konteksta izvēlnē izvēlēties punktu **Add | File...** un dialogā, izvēlēties datu avota failu **C:\Work\QualificationWorks.sdf**. Pēc šī faila ievietošanas virtuālajā katalogā, ir iespējams uzstādīt tā īpašības attiecībā pret instalācijas procesu un gala lietotāja datoru. Piemēram, īpašībā **TargetName** var norādīt faila nosaukumu gala lietotāja datorā. Lai par to pārliecinātos, īpašību logā nomainīt faila **QualificationWorks.sdf** īpašības **TargetName** vērtību uz **QualifWorksData.sdf**.

- Ar šo instalātora moduļa **MergeModule** projekts ir pabeigts - pārliecināties par tā kompilēšanas veiksmīgumu, risinājuma pārlūkā uz projekta **MergeModule** nosaukuma virsotnes konteksta izvēlnes izvēloties punktu **Build**.

Lai instalātora moduli **MergeModule** iekļautu iepriekš izveidotajā instalātora projektā **QualifWorksSetup**:

- Lai gala lietotājam būtu iespējas veikt izmaiņas datu avota failā, nepieciešams izvēlēties vietu, kurā lietotājam ir modificēšanas tiesības. Viens no šādiem katalogiem, kurā lietotājam ir šādas tiesības, instalātora projektā tiek saukts par virtuālo katalogu **User's Application Data Folder**., bet tā identifikators ir **[AppDataFolder]**. Tomēr izmantosim citu pieeju - izveidosim pielāgotu virtuālo

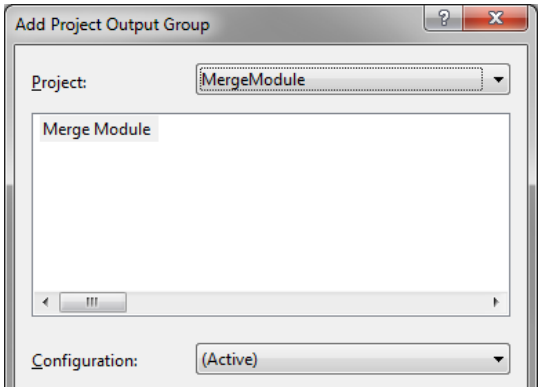
katalogu, kura noklusētā vērtība būs vienāda ar identifikatoru **[AppDataFolder]**. Šim nolūkam, izmantojot instalātora projekta failu sistēmas redaktora konteksta izvēlnes punktu **Add Special Folder | Custom Folder**, pievienot virtuālo katalogu, kuru pārdēvēt par **Data**:



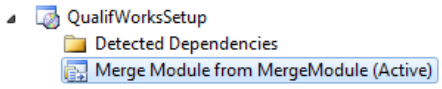
Īpašību logā šim katalogam uzstādīt, ka īpašības **Property** vērtība ir identifikators **DATASOURCE**, bet īpašības **DefaultLocation** vērtība ir **[AppDataFolder]**:

(Name)	Data
AlwaysCreate	False
Condition	
DefaultLocation	[AppDataFolder]
Property	DATASOURCE
Transitive	False

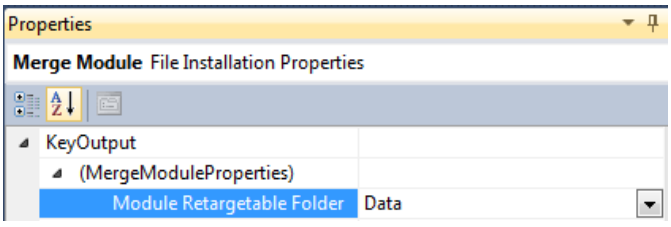
- Failu sistēmas redaktorā uz virtuālā kataloga **Data** virsotnes konteksta izvēlnes izvēlēties punktu **Add | Project Output...**, atvērsies dialogs, kurā izvēlēties projektu **MergeModule**:



- Ar pogas OK nospiešanu, instalātora moduļa **MergeModule** gala artefakts tiek iekļauts instalātora projektā **QualifWorksSetup**:



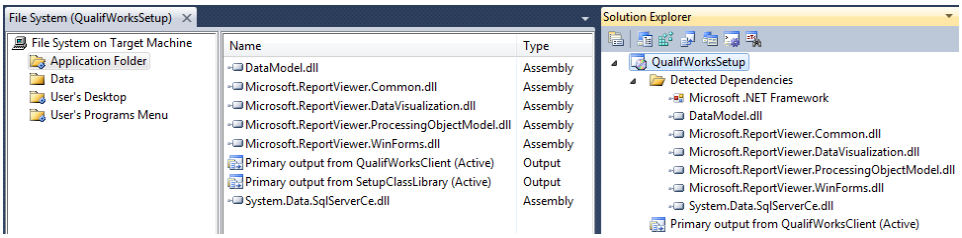
- Risinājuma pārlūka logā izvēlēties pievienotā moduļa virsotni un īpašību logā īpašībai **KeyOutput** norādīt moduļa artefaktu kopēšanas vietu uz virtuālo katalogu **Data**:



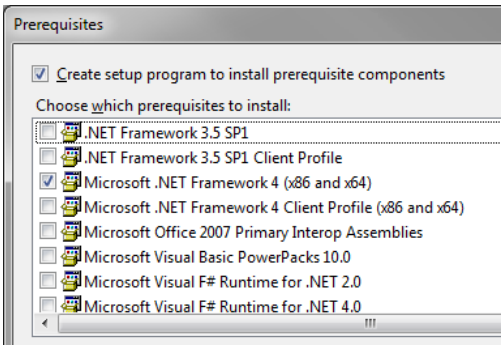
Lai instalātora projektā **QualifWorksSetup** iekļautu lietojumprogrammas projekta **QualifWorksClient** gala artefaktus:

- Izmantojot iepriekš izmantoto konteksta izvēlnes komandu **Add | Project Output...** (tikai šoreiz uz virtuālā kataloga **Application Folder** virsotnes), norādīt projektu **QualifWorksClient**, sarakstā izvēloties elementu **Primary output**. Pēc

dialoga aizvēršanas, var redzēt, ka failu sistēmas redaktora virtuālajā katalogā **Application Folder**, tiek iekļauts gan projekta gala artefakts **QualifWorksClient.exe**, gan arī visi šim artefaktam nepieciešamie faili (ieskaitot arī projektā **DataModel** definēto **DataModel.dll**):



- Veicot instalātora projekta **QualifWorksSetup** kompilēšanu, var tikt izvadīts brīdinājums, ka instalātora projekta mērķa platformas konfigurācija '.NET Framework 3.5/4 Client Profile' neatbilst instalējamās lietojumprogrammas **QualifWorksClient** mērķa platformas konfigurācijai '.NET Framework 3.5/4'. Lai šo brīdinājumu novērstu, jāatver instalātora īpašību dialogs (instalātora projekta virsotnes konteksta izvēlnes punkts **Properties**), instalātora īpašību dialogā jānospiež poga **Prerequisites...**, un priekšnosacījumu dialogā jāizvēlas atbilstošās konfigurācijas (Visual Studio 2008 versijai - .NET Framework 3.5 SP1):



Šādi tiek garantēts, ka lietojumprogramma netiks uzinstalēta, kamēr nebūs apmierināti visi iezīmētie priekšnosacījumi, instalātoram piedāvājot tos uzinstalēt pirms lietojumprogrammas instalēšanas.

Tā kā lietojumprogrammā **QualifWorksClient** pēc noklusēšanas kā datu avots tiek izmantota SQL Server Compact 3.5 datu bāze, kā priekšnosacījumu izvēlēties arī opciju **SQL Server Compact 3.5 SP2**, jo lietojumprogrammas instalēšanas brīdī, gala lietotāja datorā tā var nebūt uzstādīta.

- Risinājuma pārlūka logā aktivizēt instalātora projekta **QualifWorksSetup** virsotni un īpašību logā uzstādīt instalātora īpašības:

Īpašība:	**Vērtība:**
Author	Jūsu vārds un uzvārds.
Description	Noslēgumu darbu uzskaites sistēma.
Manufacturer	Student
ProductName	Qualification Works
Title	Qualification Works Setup

Lai lietojumprogrammu varētu ērti palaist, izveidosim tās saīsni uz ekrāna darbvirsmas:

- Atvērt instalātora projekta **QualifWorksSetup** failu sistēmas redaktoru un aktivizēt virtuālo katalogu **Application Folder**.
- Šī kataloga saturā izvēlēties elementu **Primary output from QualifWorksClient (Active)**, un tā konteksta izvēlnē, izvēlēties punktu **Create shortcut to...**.

- Pārdēvēt jaunizveidoto saīsni par **Noslēguma darbi**, un, izmantojot peli, aizvilkt to uz virtuālo mapi **User's Desktop**.

Kaut arī varētu šķist, ka ar šo instalātora projekts **QualifWorksSetup** ir veiksmīgi pabeigts un instalātors ir sagatavots, un, kaut gan tas tiks veiksmīgi nokompilēts un pēc tam, izmantojot instalātora projekta konteksta izvēlnes komandu **Install** palaists, un veiksmīgi uzinstalēts, gala lietotāja datorā piekļūšana datu avotam nebūs iespējama, jo:

1. Datu avota faila atrašanās vieta gala lietotāja datorā ir uzstādīta uz virtuālā kataloga **Data** vērtību, par ko lietojumprogrammai **QualifWorksClient.exe** nekas nav zināms;
2. Projektā **DataModel** informācija par datu avotu tika norādīta kā savienojuma virkne ar vērtību **C:\Work\QualificationWorks.sdf**, bet instalātora moduļa projektā **MergeModule** tika norādīts, ka gala lietotāja datorā datu avota fails ir jākopē ar nosaukumu **QualifWorksData.sdf**.

Lai novērstu šo nepilnību, izmantosim lietojumprogrammas konfigurācijas faila **QualifWorksClient.exe.config** iespēju ārēji pārdefinēt lietojumprogrammas uzstādījumos saglabāto datu avota savienojuma virkni. Tā rezultātā, palaižot lietojumprogrammu, tiks konstatēts, ka tajā pašā katalogā atrodas arī konfigurācijas fails, līdz ar to uzstādījumu vērtība tiks ņemta no šī faila - ja šāda faila nebūs, tad tiks ņemta tā, kas tika norādīta izstrādes vidē un ir saglabāta artefaktā **DataModel.dll**.

Darbam ar konfigurācijas failiem .NET bibliotēkas vārdu telpā **System.Configuration**, ir pieejamas speciālas klases: **Configuration**, **ConfigurationManager** un **ConnectionStringSettings**. Izmantojot tās, ir iespējams dinamiski izveidot un modificēt lietojumprogrammas konfigurācijas failu, līdz ar to saglabājot tajā faktisko gala lietotāja datora kataloga un faila nosaukuma vērtības.

Lai ar minētajām klasēm šo funkcionalitāti realizētu, izmantosim instalātora iespēju izsaukt speciālus apakšinstalātorus, kas nodrošina patvaļīga koda izpildi instalēšanas laikā. Šādu instalātoru izveidi nodrošina .NET bibliotēkas klase **System.Configuration.Install.Installer**.

Šim nolūkam:

- Risinājumam **QualificationWorks** pievienot jaunu projektu, izmantojot sagatavi **Class Library**, kuru nosaukt par **SetupClassLibrary**.
- Šajā projektā izdzēst noklusēto klasi **Class1**.
- Projektam pievienot jaunu klasi, par pamatu izmantojot sagatavi **Installer Class**, kuru nosaukt par **ConnectionInstaller.cs**.
- Atvērsies jaunās klases redaktors (patiesībā šī klase ir arī komponents), kurā izvēlēties saiti "click here to switch to code view", un klases pirmkodu papildināt ar šādu realizāciju:

```
namespace SetupClassLibrary
{
    [RunInstaller(true)]
    public partial class ConnectionInstaller
                                        : System.Configuration.Install.Installer
    {
        public ConnectionInstaller()
        {
            InitializeComponent();
        }

        private const int AppTarget = 0;
        private const int Name = 1;
        private const int ConnString = 2;
        private const int DataProvider = 3;

        private const string SqlServerCe = "Microsoft.SqlServerCe.Client.3.5";

        public override void Install(IDictionary stateSaver)
```

```
        {
            try
            {
                // Obligāti vispirms jāizsauc tāda paša nosaukuma
                // bāzes klases metode:
                base.Install(stateSaver);

                string[] args = new string[4];

                // Instalātoram nodoto parametru nolasīšana:
                args[Name] =
               "DataModel.Properties.Settings.QualificationWorksConnectionString";

                args[AppTarget]
                   = Context.Parameters["AppTarget"].Replace("\\\\", "\\")
                   + "QualifWorksClient.exe";

                args[DataProvider] = Context.Parameters["DataProvider"];
                if (args[DataProvider].Length == 0)
                    args[DataProvider] = SqlServerCe;

                args[ConnString]
                         = "Data Source="
                         + Context.Parameters["DataSource"].Replace("\\\\", "\\");
                if (args[DataProvider] == SqlServerCe)
                    args[ConnString] += "QualifWorksData.sdf";

                //if (MessageBox.Show(args[AppTarget] + "\n" + args[Name] + "\n" +
args[ConnString] + "\n" + args[DataProvider], "", MessageBoxButtons.OKCancel) ==
DialogResult.Cancel)
                //    throw new Exception("Installation cancelled.");

                // Konfigurācijas faila savienojuma virknes sagatavošana:
                System.Configuration.Configuration config
                    = ConfigurationManager.OpenExeConfiguration(args[AppTarget]);
                ConnectionStringsSection csSection = config.ConnectionStrings;
                ConnectionStringSettings csSettings
                                       = csSection.ConnectionStrings[args[Name]];

                if (csSettings == null)
                {
                    csSettings = new ConnectionStringSettings(args[Name], "");
                    csSection.ConnectionStrings.Add(csSettings);
                }

                csSettings.ConnectionString = args[ConnString];
                csSettings.ProviderName = args[DataProvider];

                // Saglabāšana konfigurācijas failā:
                config.Save(ConfigurationSaveMode.Modified);
            }
            catch (Exception ex)
            {
                MessageBox.Show(
                    string.Format("Kļūda metodē Install(): {0}, {1}"
                                , ex.Source, ex.Message));
                throw;
            }
        }
    }
}
```

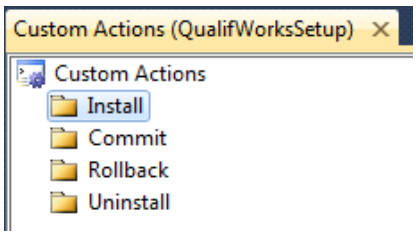
- Lai novērstu konfigurācijas faila klašu neatpazīšanas kļūdas, risinājuma pārlūka logā projekta **SetupClassLibrary** atsauču virsotnei **References** pievienot šādus .NET klašu avotus:
    - **System.Configuration**
    - **System.Windows.Forms**

  Pēc tam, lai iekļautu neatpazīto klašu vārdu telpas, pirmkoda **ConnectionInstaller.cs** redaktorā izmantot komandu **Resolve**,.

- Nokompilēt klašu projektu **SetupClassLibrary**, lai pārliecinātos, ka tajā nav kļūdas.

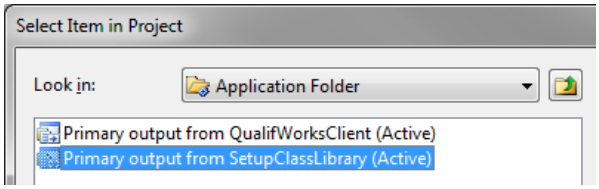
Lai no instalātora projekta **QualifWorksSetup** varētu izsaukt projekta **SetupClassLibrary** gala artefaktu **SetupClassLibrary.dll**, jārīkojas šādi:

- Instalātora projektam **QualifWorksSetup** pievienot projekta **SetupClassLibrary** gala artefaktus (**Add | Project Output...**).
- Risinājuma pārlūkā projekta **QualifWorksSetup** virsotnes konteksta izvēlnē, izvēlēties punktu **View | Custom Actions**, atvērsies pielāgojamo darbību redaktora logs:



  Šajā redaktorā iespējams piesaistīt papildus darbības, kas tiks automātiski aktivizētas atbilstošajos instalēšanas/atinstalēšanas posmos.

- Uz virsotnes **Install** konteksta izvēlnes, izvēlēties punktu **Add Custom Action...** un dialogā pāriet uz virtuālo mapi **Application Folder**, kurā izvēlēties elementu ar projekta **SetupClassLibrary** nosaukumu:



- Pēc dialoga aizvēršanas, pielāgojamo darbību virsotnei **Install** tiks pievienota jauna apakšvirsotne **Primary output from SetupClassLibrary (Active)**, kas instalēšanas gaitā ielādēs artefaktā **SetupClassLibrary.dll** esošo klasi **ConnectionInstaller**, izveidos tās instanci, un izsauks instances metodi **Install()**.

  Bet, tā kā lai izpildītu šo metodi, tai nepieciešama papildus informācija, proti, kur faktiski tiek kopēts datu avota fails un pati lietojumprogramma, tad šo informāciju nepieciešams nodot, izmantojot pielāgotās darbības īpašību **CustomActionData**.

  Norādīt tajā šādu vērtību:

  /AppTarget="[TARGETDIR]\" /DataSource="[DATASOURCE]\" /DataProvider="Microsoft.SqlServerCe.Client.3.5"

  Redzams, ka šādi tiek izmantoti instalātorā definētie identifikatori **TARGETDIR** un **DATASOURCE**, kā arī citi parametri, kuru vērtības ir teksta virknes.

  Izsaucot pielāgoto darbību, norādītie identifikatori tiks aizvietoti ar to faktiskajām vērtībām. Savukārt nodotajiem parametriem pašā pielāgotajā instalātorā var piekļūt, piemēram, ar šādu izteiksmi: **Context.Parameters["AppTarget"]**. Ja sagaidāmais parametrs netiek nodots, tiek izraisīta izņēmuma situācija - tāpēc pielāgotā instalātora metodē **Install()** tiek izmantots **try-catch** mehānisms, kas atvieglo tā atkļūdošanas procesu.

- Pielāgoto darbību virsotnei **Uninstall** patstāvīgi pievienot projekta **SetupClassLibrary** artefaktu - tas nepieciešamas, lai šī pati pielāgotā darbība tiktu izsaukta arī atinstalēšanas laikā. Atinstalēšanas laikā tiks izsaukta klases **ConnectionInstaller** metode **Uninstall()**, bet tā kā tā nav definēta, tad tiks izpildīta bāzes klases **Installer** tāda paša nosaukuma metode.

Ar šo var uzskatīt, ka sākotnēji izvirzītās prasības pret lietojumprogrammas instalātoru ir izpildītas - instalātora projektu var nokompilēt, un pēc tam pārbaudīt, izsaucot tā instalācijas komandu:

- Instalācijas programmas vedņa solī, kur jānorāda instalācijas katalogs, noklusētā kataloga **C:\Program Files\Student\Qualification Works** vietā norādīt **C:\Work\Student\Qualification Works**, un atstāt noklusēto instalācijas variantu, kurā instalācija tiks veikta tikai tekošajam lietotājam (**Just me**).

Ja instalācijas process beidzās veiksmīgi, pārliecināties, ka uz darbvirsmas ir izveidota saīsne **Noslēguma darbi**, caur kuru ar peles dubultklikšķi var atvērt lietojumprogrammu, kura veiksmīgi ir piekļuvusi datu avotam. Pārliecināties, ka lietojumprogrammas instalācijas katalogā ir izveidots tās konfigurācijas fails **QualifWorksClient.exe.config**, un iepazīties ar tā saturu un formātu. Noskaidrot, kur patiesībā tiek iekopēts datu avota faila **QualifWorkData.sdf**.

Veikt lietojumprogrammas atinstalēšanu, un pārliecināties, ka visi uzinstalētie artefakti (darbvirsmas saīsne, faili un katalogi), tiek korekti izdzēsti.