Customization AutoCAD2008

Esa Salmikangas
Senior Lecturer

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JYVÄSKYLÄ UNIVERSITY OF APPLIED SCIENCES

Jyväskylä University
Of Applied Sciences

Getting Acquainted

The lecturer: Esa Salmikangas

- More than ten years worked in SW industry
  - Customization AutoCAD
  - Database, Workgroup, Business Intelligence
- Eight year Senior Lecturer in University of Applied Sciences (former Polytechnic)
- Very practical and pragmatic way of thinking in software engineering

Who Are You?

- You interests?

Course Objectives

is give:

- Give an example what can be done

is to understand:

- the fundamentals of the AutoCAD Customization .NET API
- the basics of the Lisp, VBA and AutoCAD .NET API

What it is not:

- Teach you .NET framework or C# , VB programming language
- Give you complete of coverage of all API functions

What is AutoCAD? Why?

- **Fact #1:**
  - AutoCAD is a CAD software application for 2D and 3D design and drafting, developed and sold by Autodesk, Inc.
- **Fact #2:**
  - One of the leading and most used 2D CAD software in the world
  - De facto-standard in Scandinavia in CAD
- **Fact#3:**
  - AutoCAD is open, so it has many, many possibilities to customize.

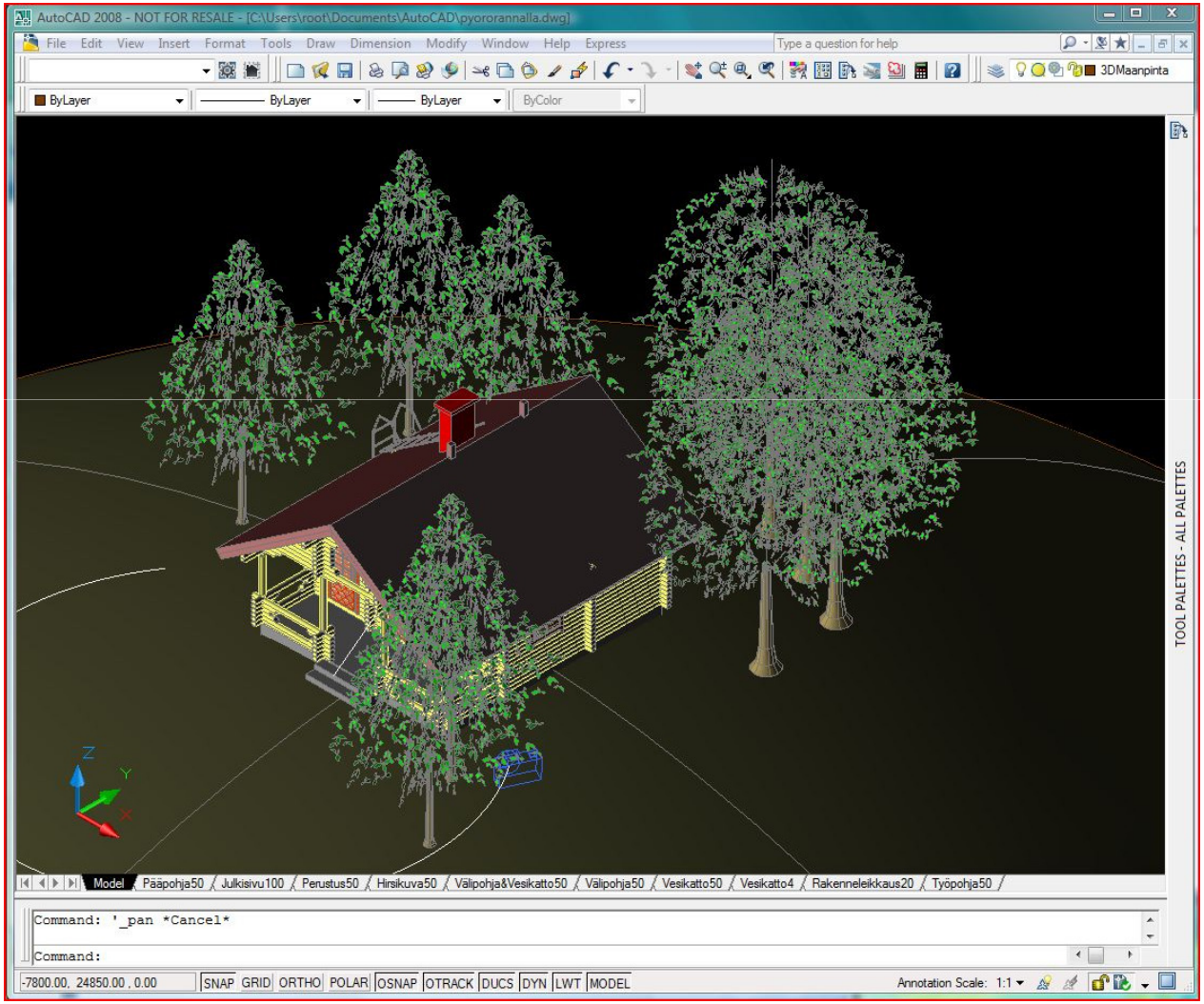
AutoCAD releases history
http://myfeedback.autodesk.com/history/autocad_release_history.htm

Customization of AutoCAD

- In the beginning Autodesk just made general CAD and customers self or so called third-part-companies made tailoring for different needs of customers
- In time being Autodesk has produced more high-end applications. Now there is:
  - The *Platform Solutions and Emerging Business* division develops and manages Autodesk's flagship product, AutoCAD, AutoCAD LT, Autodesk's Geospatial solutions, Plant solutions, Extended Design offerings such as Design Review, Content and Search solutions, Autodesk Labs, and worldwide engineering.
  - The *Manufacturing Solutions Division* develops and manages Autodesk Inventor Series, Autodesk Inventor Professional, AutoCAD Mechanical and Autodesk Vault.
  - The *Architecture Engineering and Construction* division develops and manages AutoCAD Architecture (Old name - Architectural Desktop), Autodesk Building Systems, AutoCAD Revit Architecture (Old name - Revit Building), AutoCAD Revit Structure, AutoCAD Revit MEP (Old name - Revit Systems), and AutoCAD Civil 3D.

AutoCAD 2008

AutoCAD 2008 (R17.1) - March 2007



More than two dozen ways to customize AutoCAD…

- **some of these may vary, depending on the version of AutoCAD you are working with:**
  - ADI - Autodesk Device Interface and plotting formats (replaced by HDI in AutoCAD 2000)
  - ADS - AutoCAD Development System (no longer available as of AutoCAD 2000).
  - API - Applications programming interface
  - ASI - AutoCAD SQL Interface
  - CUI – new in AutoCAD 2006 Customizable user interface
  - DCL - Customizable dialog boxes
  - DDE - Dynamic Data Exchange
  - DIESEL - Direct Interpretively Evaluated String Expression Language
  - DVB – VBA project
  - DWG - DraWinG; create custom symbols and user-defined objects.
  - DXB - Drawing Interchange Binary
  - DXF - Drawing Interchange Format
  - DXFIX - Drawing translation (no longer available as of AutoCAD 2000)
  - HLP and AHP - Customizable help (no longer available as of AutoCAD 2000)
  - INI - Toolbar macros and initialization files (no longer available as of AutoCAD 2000)
  - LIN - Customizable linetypes
  - LSP - AutoLISP
  - MNU - Customizable menu and tablet, popdown, cursor, and icon menus
  - ObjectARX - AutoCAD Runtime Extension
  - OLE - Object linking and embedding
  - PAT - Customizable hatch patterns
  - PGP - Program parameter files
  - RPF - Raster pattern file
  - SCR - Script files
  - SHP - Customizable text fonts
  - SHX - Shapes
  - SLD and SLB - Slides and slide libraries
  - VBA - Visual Basic for Applications
  - VLISP - Visual Lisp

What to use? All?

- **Some of theses are designed for end-users:**
  - like toolbar macros, menus, and AutoLISP routines
- **Others are meant for professional programmers, like :**
  - DVB, ASI and ObjectARX,
- **In between, there are dozens of other customization possibilities:**
  - like hatch patterns and DIESEL programming, that some enthusiastic users enjoy tinkering with.

More than two dozen, my “favorites”…

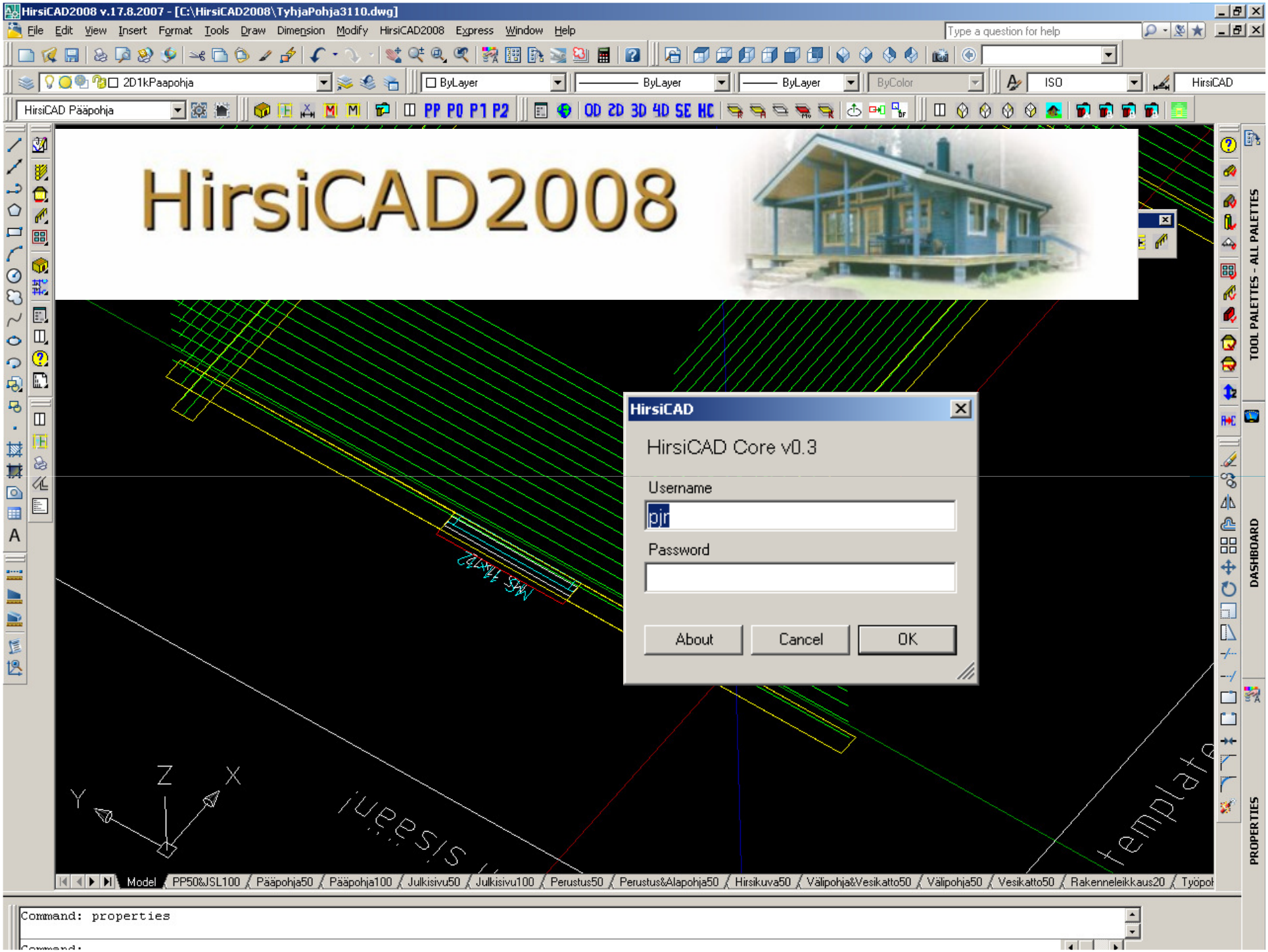
- ADI - Autodesk Device Interface and plotting formats (replaced by HDI in AutoCAD 2000)
- ADS - AutoCAD Development System (no longer available as of AutoCAD 2000).
- API - Applications programming interface
- ASI - AutoCAD SQL Interface
- **CUI – new in AutoCAD 2006 Customizable user interface**
- DCL - Customizable dialog boxes
- DDE - Dynamic Data Exchange
- DIESEL - Direct Interpretively Evaluated String Expression Language
- **DVB – VBA project**
- DWG - DraWinG; create custom symbols and user-defined objects.
- DXB - Drawing Interchange Binary
- DXF - Drawing Interchange Format
- DXFIX - Drawing translation (no longer available as of AutoCAD 2000)
- HLP and AHP - Customizable help (no longer available as of AutoCAD 2000)
- INI - Toolbar macros and initialization files (no longer available as of AutoCAD 2000)
- LIN - Customizable linetypes
- LSP - AutoLISP
- MNU - Customizable menu and tablet, popdown, cursor, and icon menus
- **ObjectARX - AutoCAD Runtime Extension**
- OLE - Object linking and embedding
- PAT - Customizable hatch patterns
- PGP - Program parameter files
- RPF - Raster pattern file
- SCR - Script files
- SHP - Customizable text fonts
- SHX - Shapes
- SLD and SLB - Slides and slide libraries
- VBA - Visual Basic for Applications
- VLISP - Visual Lisp

Plus:
- DB: Access or SQL Server
- XML files for configuration

An example: HirsiCAD 2008

Customized AutoCAD application

for design loghouses


HirsiCAD2008 v.17.8.2007 - [C:\HirsiCAD2008\TyhjaPohja3110.dwg]
File Edit View Insert Format Tools Draw Dimension Modify HirsiCAD2008 Express Window Help
Type a question for help
2D1kPaapohja
ByLayer
ByLayer
ByLayer
ByColor
ISO
HirsiCAD
HirsiCAD Pääpohja
PP P0 P1 P2
OD 2D 3D 4D SE HC
HirsiCAD2008
HirsiCAD
HirsiCAD Core v0.3
Username
pjr
Password
About
Cancel
OK
TOOL PALETTES - ALL PALETTES
DASHBOARD
PROPERTIES
Model
PP50&JSL100
Pääpohja50
Pääpohja100
Julkisivu50
Julkisivu100
Perustus50
Perustus&Alapohja50
Hirsikuva50
Välipohja&Vesikatto50
Välipohja50
Vesikatto50
Rakenneleikkaus20
Työpoh
Command: properties

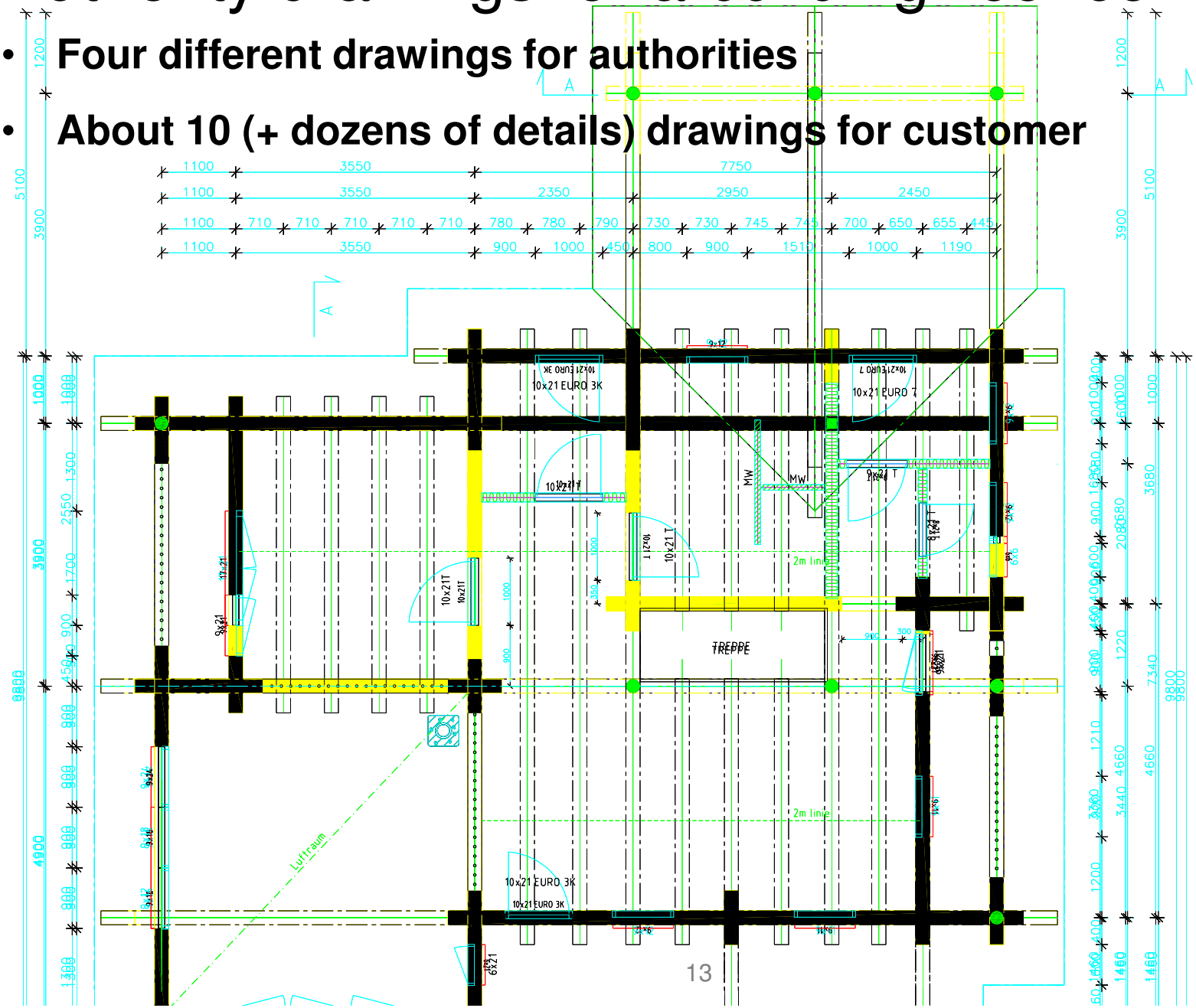
What does it?

- **Main goals:**
  - Helps designers in routines
    - design&drawing time for one log-house is approximately 24 hours, in demanding project more
    - Produce some material data automatically
  - Drawings for authorities
  - Timber handling is automated,

    so we HirsiCAD produce information for steering saws, drills etc
  - Uses data from database (sales information)
- **Secondary goals:**
  - 3D (note it is only extra)
  - Much more

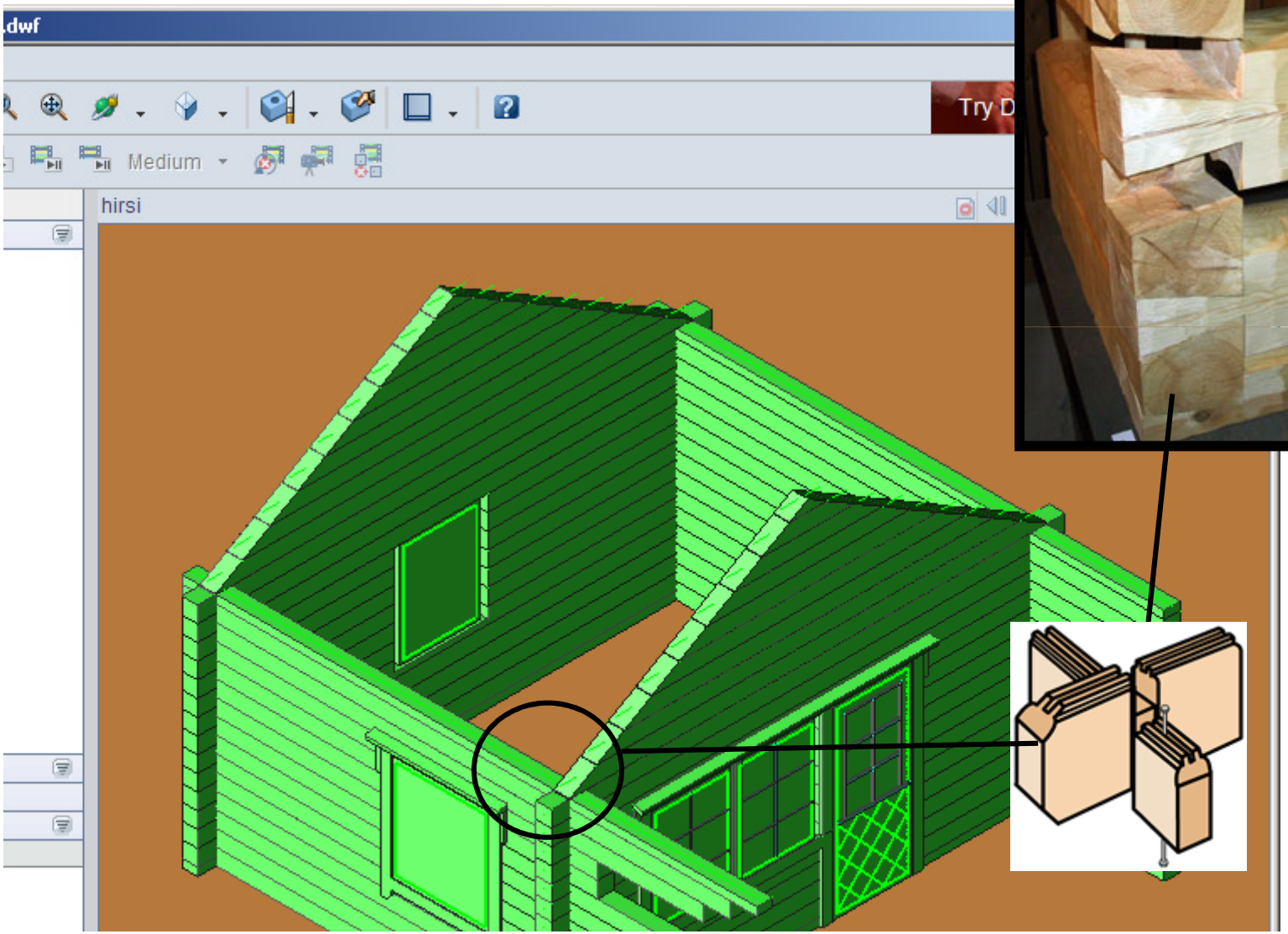
Authority drawings for a building licence

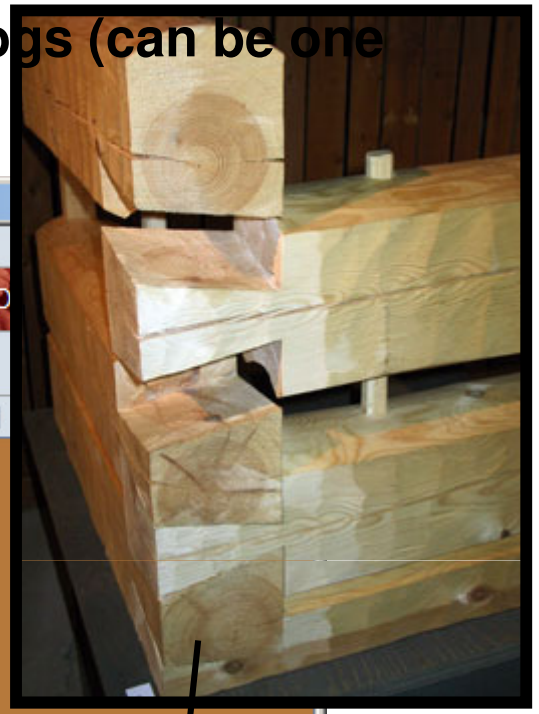
- **Four different drawings for authorities**
- **About 10 (+ dozens of details) drawings for customer**



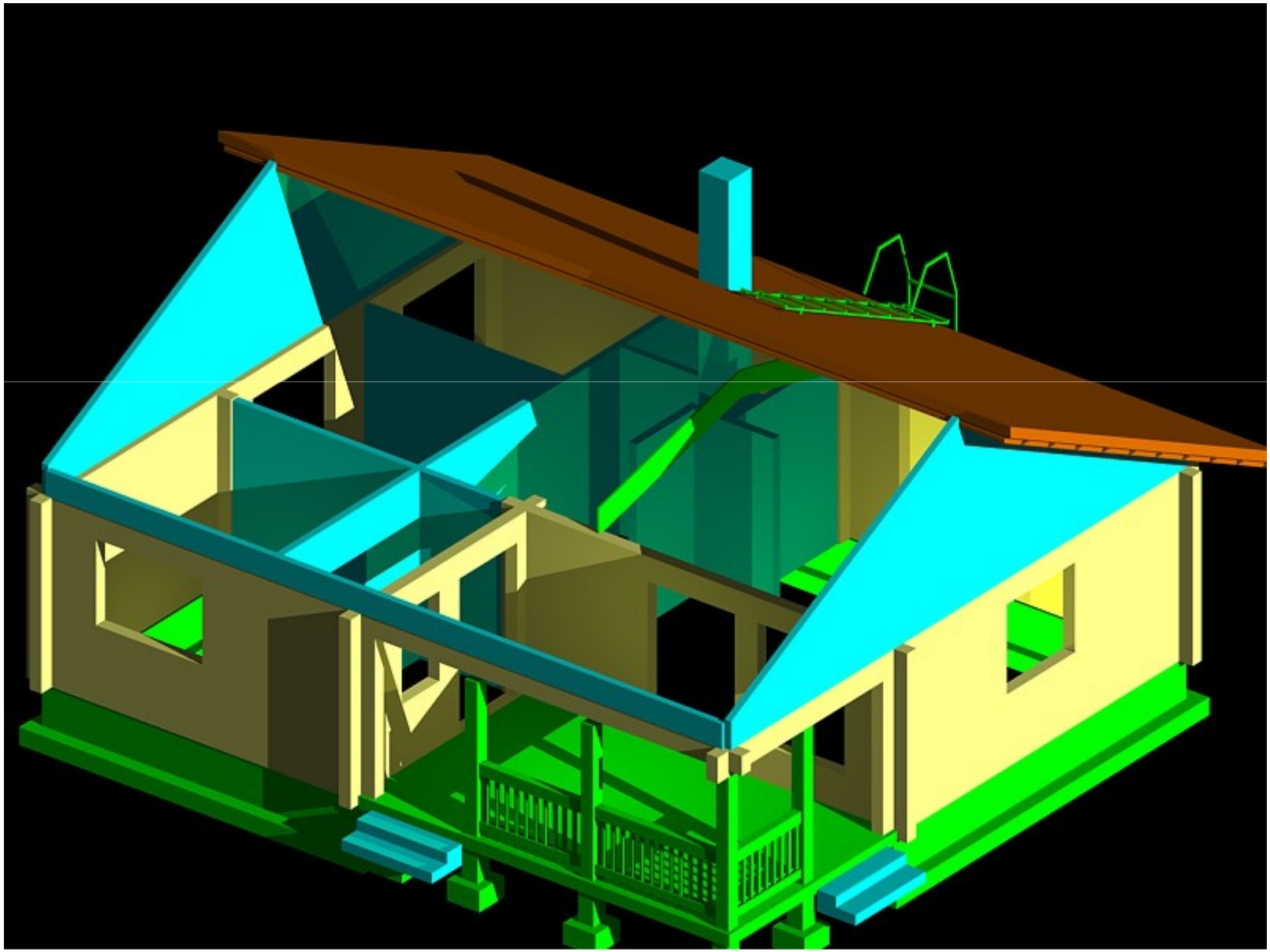
Automation

- **Produce automatically all information how logs (can be one thousand different) must be machined**



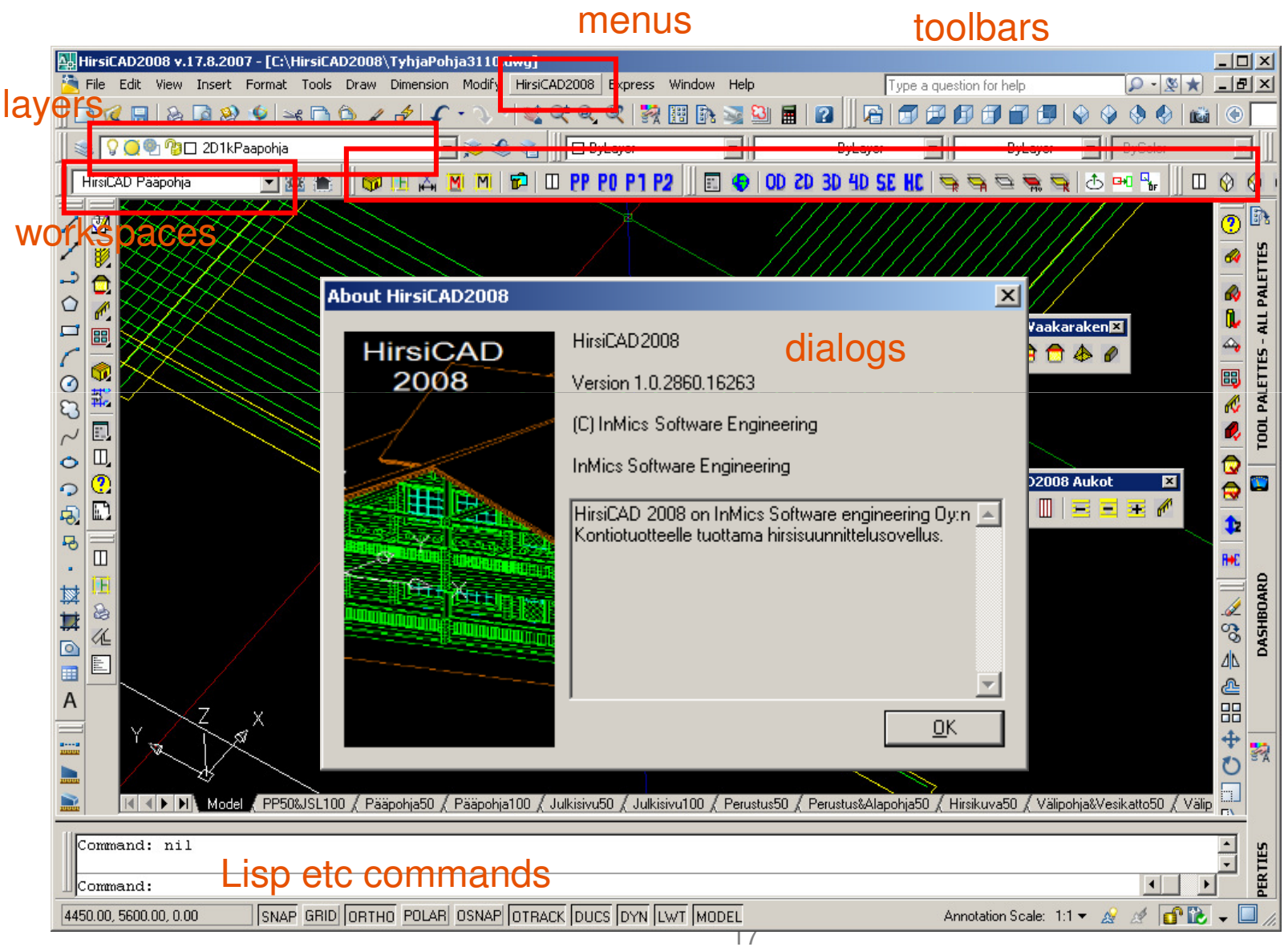


3D is something extra



Rendering has come better…

Customized User Interface here in finnish

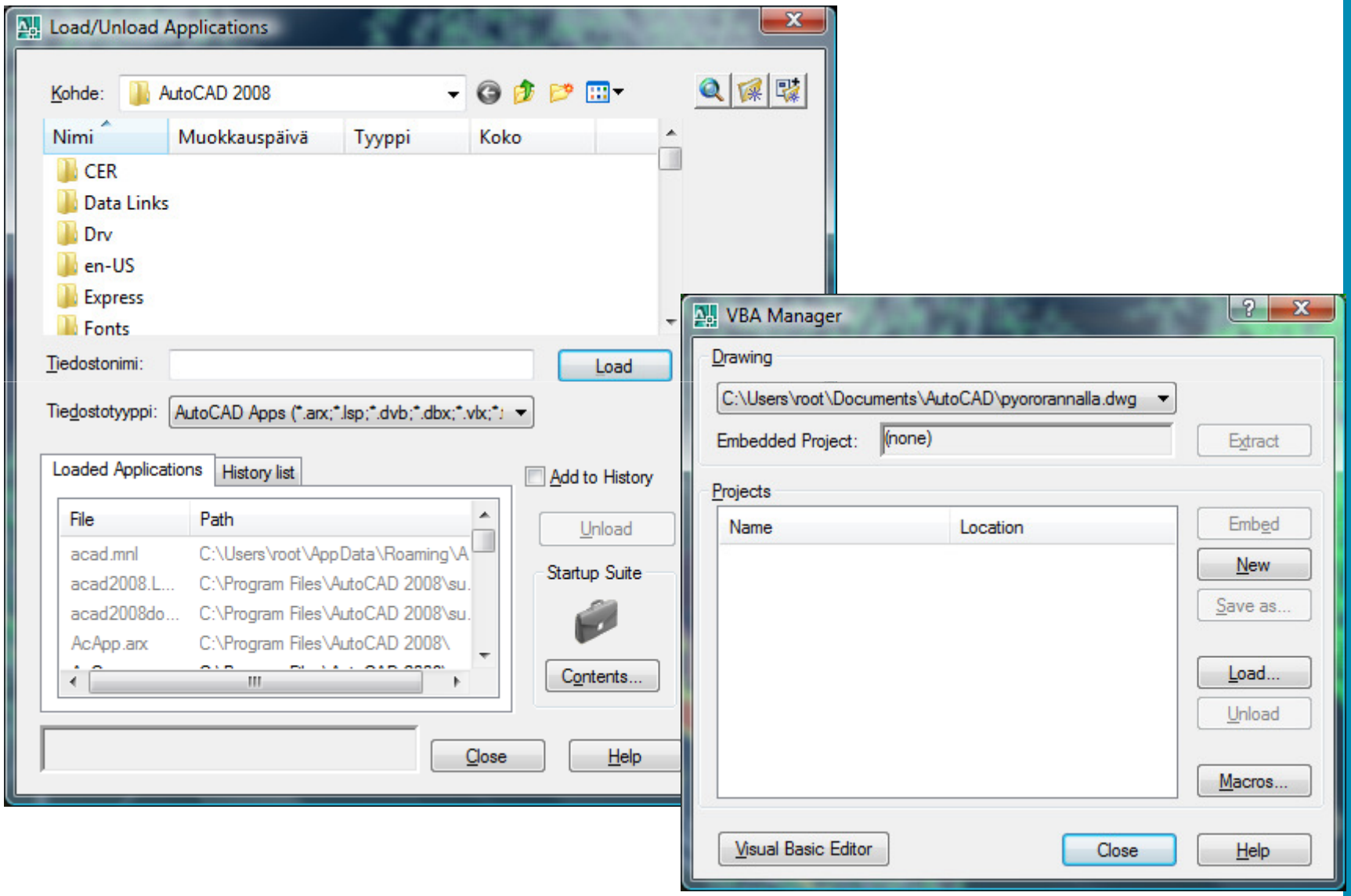


Jyväskylä University of Applied Sciences

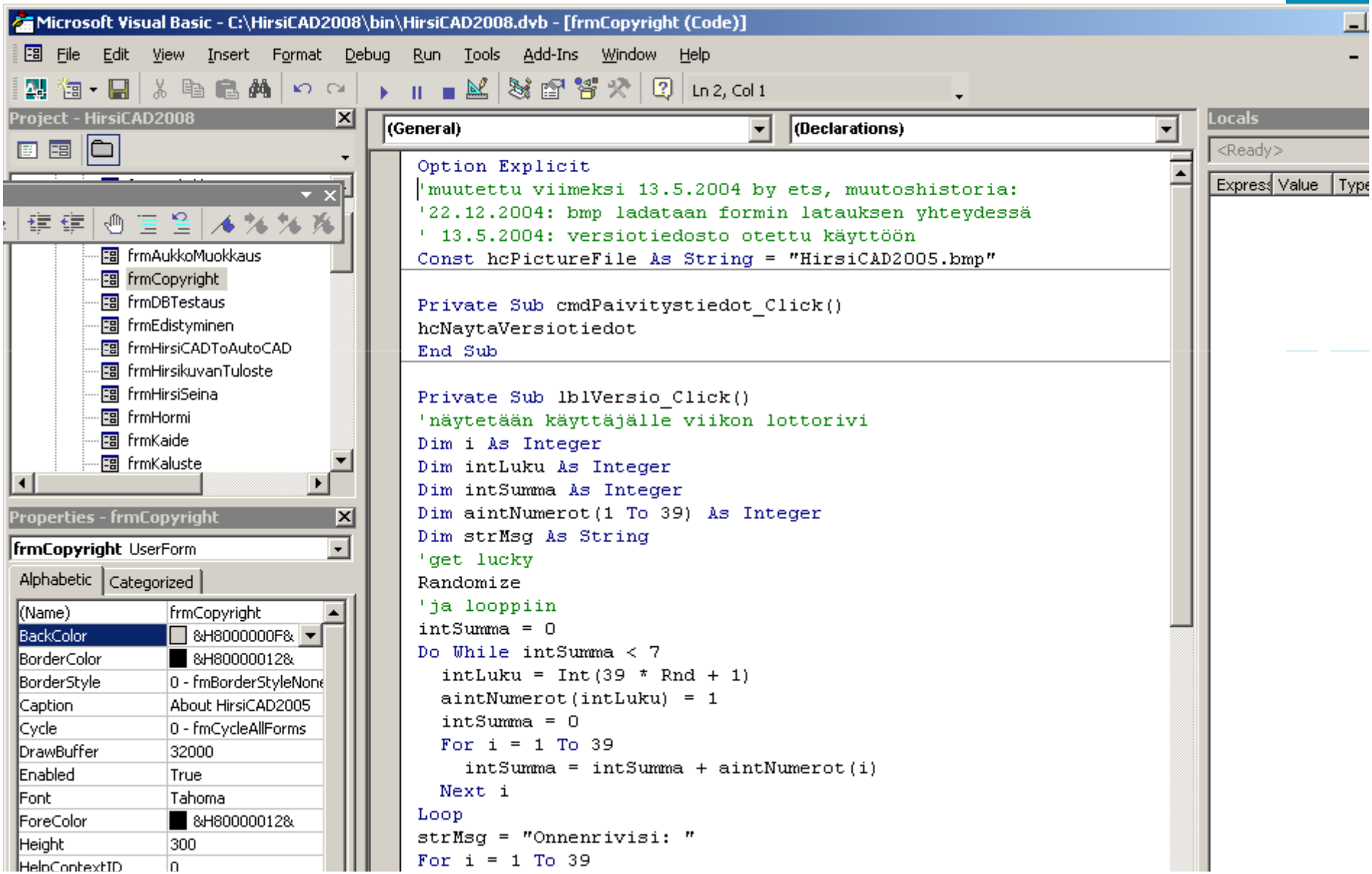
HirsiCAD 2008: programs

- **AutoLISP**
- **VBA**
  - all main functionality
  - all dialogs
  - lot of classes
- **.NET**
  - Core
  - automatic upgrade
  - Before with VB6 DLL

Load Applications & VBA Manager



VBA Visual Basic for Application



About Visual Basic for Applications

- **VBA is an implementation of Microsoft's Visual Basic, an event driven programming language and associated integrated development environment (IDE)**
  - built into most Microsoft Office applications.
  - as well as being at least partially implemented in some other applications such as AutoCAD, WordPerfect and ESRI ArcGIS.
  - can be used to control almost all aspects of the host application, including manipulating user interface features, such as menus and toolbars, and working with custom user forms or dialog boxes.
- **VBA itself is an interpreted language.**
  - As its name suggests, VBA is closely related to Visual Basic,
  - but can normally only run code from within a host application rather than as a standalone application.
  - It can, however, be used to control one application from another using OLE Automation.
- **VBA is functionally rich and extremely flexible but it does have some important limitations,**
  - including limited support for function pointers which are used as callback functions in the Windows API.
  - It has the ability to use (but not create) (ActiveX/COM) DLLs, and later versions add support for class modules.

About AutoLISP/VisualLISP

- **In the beginning there were only scripts in AutoCAD**
- **The language was added to AutoCAD in Version 2.18 in January 1986,**
  - continued to be enhanced in successive releases up to Release 13 in February 1995.
  - After that, its development was neglected by Autodesk in favor of more fashionable development environments. However, it has remained AutoCAD's primary user customization language.
- **AutoLISP → Visual LISP, sold as an add-on to AutoCAD 14 released in May 1997.**
  - It was incorporated into AutoCAD 2000 released in March 1999, as a replacement for AutoLISP. Since then Autodesk has chosen to halt major enhancements to Visual LISP in favor of focusing more effort on **VBA** and **ObjectARX**.

Programming AutoCAD with .NET

.NET Overview

- What is .NET?
- Benefits of programming in .NET
- Important Concepts

.NET Overview

What is .NET?

- Microsoft's Technology of a Web based infrastructure
  - Seamless interaction between applications and the Internet
  - Access information across anytime, anywhere from any device

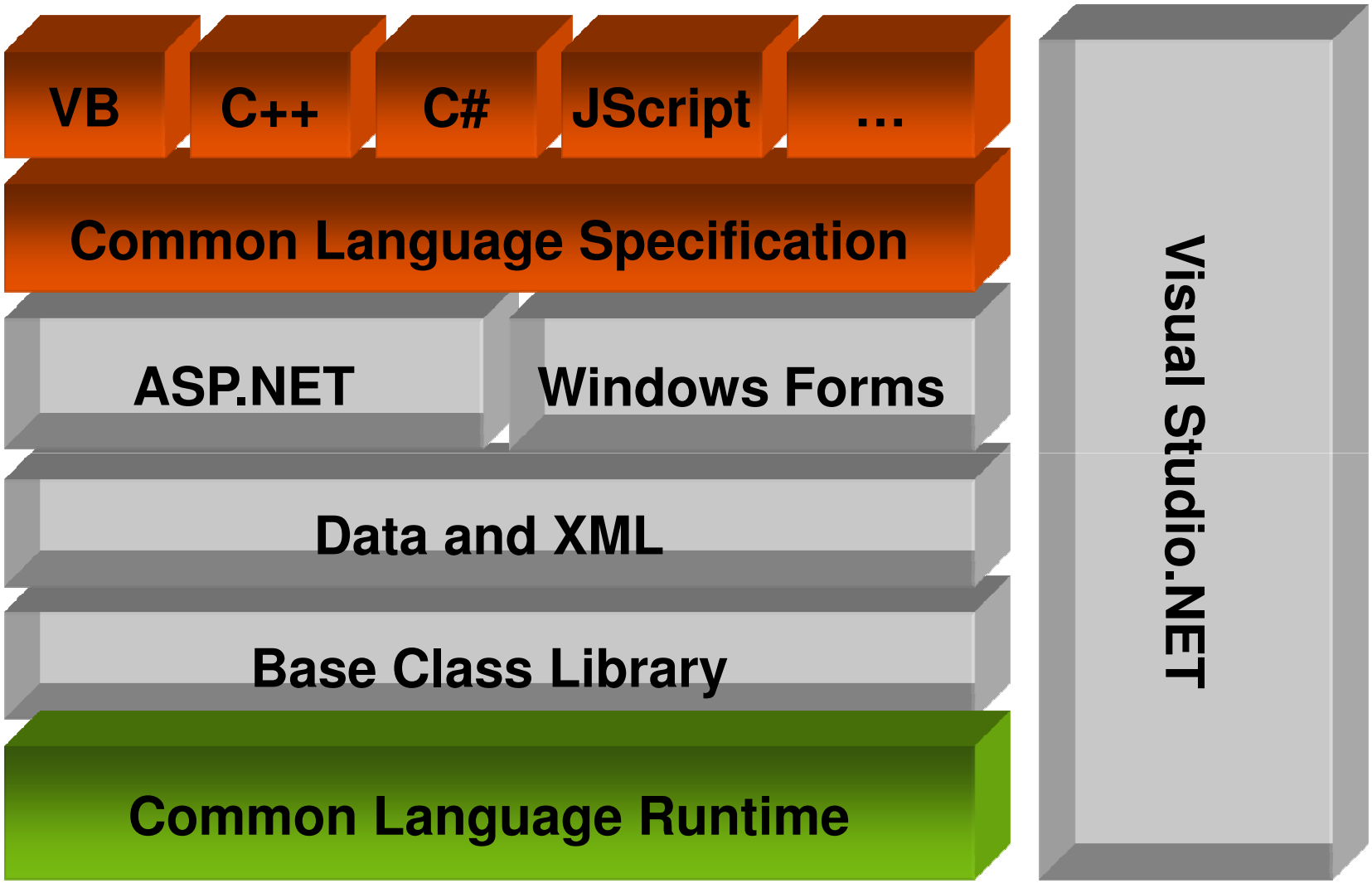
.NET Overview

What is .NET?

Components of .NET

- The .NET Framework used for building and running all kinds of software, including Web-based applications, smart client applications, and XML Web Services
- Developer tools such as Microsoft Visual Studio .NET
- A set of servers that integrate, run, operate, and manage Web services and Web-based applications

.NET Framework



.NET Overview

NET Framework

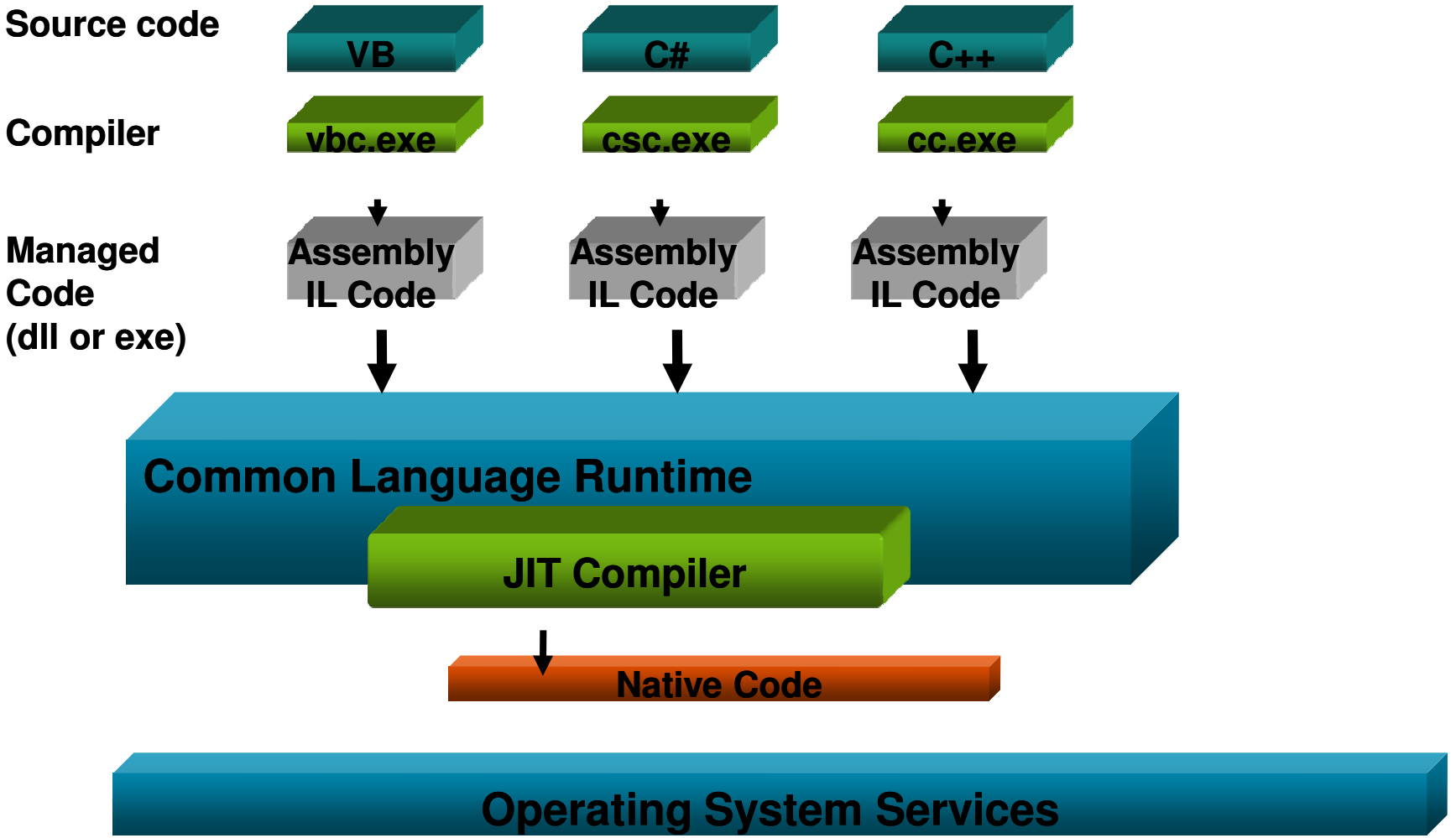
- Common Language Runtime (CLR)
  - Object-Oriented programming environment
  - Common execution environment for .NET applications

  -Similar to Java VM – but with much stronger interoperability

- Framework Class Library (FCL)
  - Object Oriented Collection of re-usable types

(Source: MSDN)

CLR Execution Model



What is Microsoft .NET?

What we know from our experience so far…

Intelligent symbolic representation

- Mature language constructs
  - Collections, Events, Delegates
- Common programming pitfalls addressed
  - Memory management, consistent Exception handling, unified strings

Source and binary inter-module communication

- goes beyond C++ and COM
  - Meta data allows design- and run-time object usage and extension

Programming style

- Multiple supported languages – Choose your weapons

.NET Overview

- What is .NET?
- Benefits of programming in .NET
- Important Concepts

.NET Overview

Benefits of programming in .NET

- Consistent Object Oriented Development platform
- Automatic memory management – Garbage collection
- Support for multiple languages

(Source: MSDN)

.NET Overview

Consistent Object Oriented Development platform

Everything you see can be treated as an Object!

```
Dim myLine As New Line()
myLine.StartPoint = New Point3d(0, 0, 0)
myLine.EndPoint = New Point3d(10, 10, 0)
myLine.GetClosestPointTo(New Point3d(5, 5.1, 0), False)

Dim x as Interger = 7
Dim s as String = x.ToString()
```

- Objects are instances of a ***Type*** or ***Class*** (for example **myLine** is an object and **Line** is a type)
- Objects have properties such as **StartPoint**, and methods such as **GetClosestPointTo()**

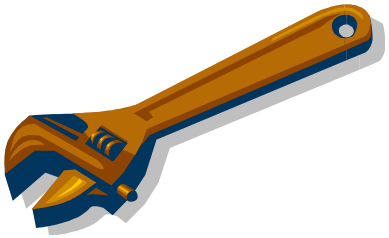
.NET Overview

Consistent Object Oriented Development platform

Mature API Constructs

What's wrong with this function?

```
int acedSSGet(const char * str,
   const void * pt1,
   const void * pt2,
   const struct resbuf * filter,
   ads_name ss);
```



.NET Overview

Consistent Object Oriented Development platform

Mature API Constructs

Some 6 new classes defined to encapsulate acedSSGet()

```
Dim values(2) As TypedValue

‘Define the selection criteria
values(0) = New TypedValue(DxfCode.Start, “Circle”)
values(1) = New TypedValue(DxfCode.Color, 1)

Dim selFilter As New SelectionFilter(values)
Dim selOpts As New PromptSelectionOptions()
selOpts.AllowDuplicates = True

‘Run the selection
Dim res As PromptSelectionResult = Editor.GetSelection(selOpts, selFilter)
```

.NET Overview

Benefits of programming in .NET

- Consistent Object Oriented Development platform
- Automatic memory management (Garbage collection) and consistent exception handling
- Support for multiple languages

.NET Overview

Benefits of programming in .NET

Automatic memory management

- Old Way (C++) - Potential for memory leaks!

```
char *pName=(char*)malloc(128);
strcpy(pName,"Hello");
//...
free(pName);
```

- New Way - .NET
  - C++ - `String *pName=new String("Hello")`
  - VB - `Dim Name As String = "Hello"`
  - C# - `String Name="Hello";`
  - `// Garbage collection handles deallocation; no 'delete'!`

.NET Overview

Benefits of programming in .NET

Consistent exception handling

- Old Way – VB: Can be very confusing and problematic!

```
On Error GoTo UnexpectedError
Dim x As Double=10/0 '…error!
UnexpectedError:
MsgBox Str$(Err.Number)
```

- New – VB .NET

```
Try
        Dim x As Double=10/0 '…error which throws exception
Catch
        '…what happened?  Division by Zero!
Finally
        '…cleanup - do this either way
End Try
```

.NET Overview

Benefits of programming in .NET

- Consistent Object Oriented Development platform
- Automatic memory management (Garbage collection) and consistent exception handling
- Support for multiple languages

.NET Overview

Benefits of programming in .NET

Support for multiple languages

- C#, VB most commonly used

- Can interop between code written in different languages. For example, a class written in C# can be inherited from a class written in VB! In fact, AutoCAD's managed assemblies are written using managed C++ which you will access from VB.NET.

.NET Overview

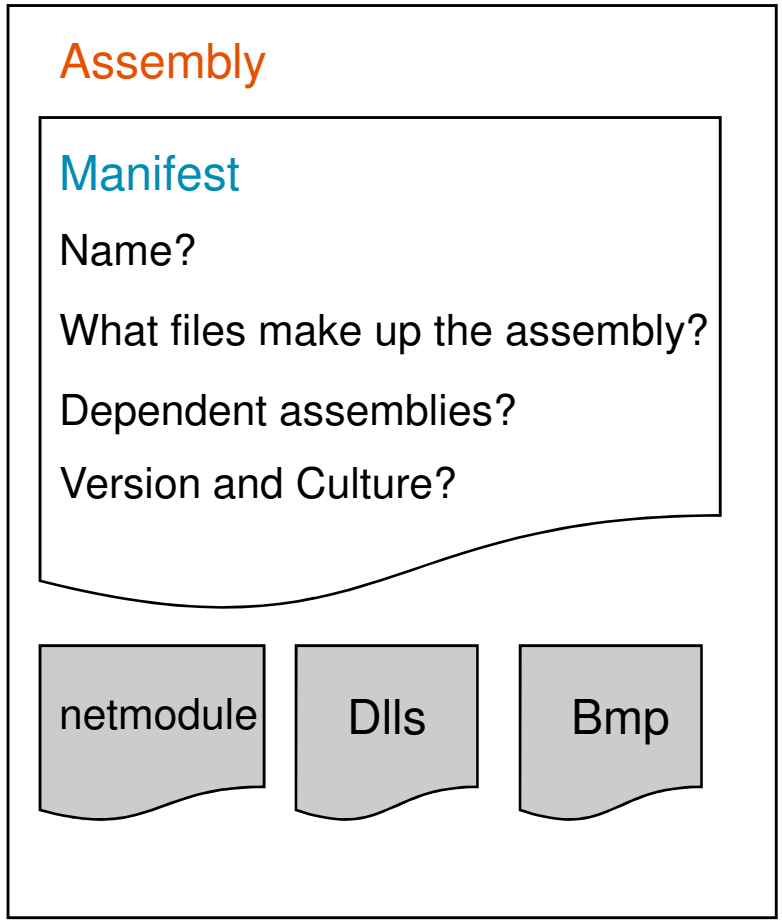
- What is .NET?
- Benefits of programming in .NET
- Important Concepts

.NET Overview

Important Concepts

Assemblies

- **Fundamental unit of deployment and execution in .NET**
  - Contains a manifest that describes the assembly
- **Boundary for code execution and access permission**



(Source: MSDN)

Class Agenda

Lectures and Labs

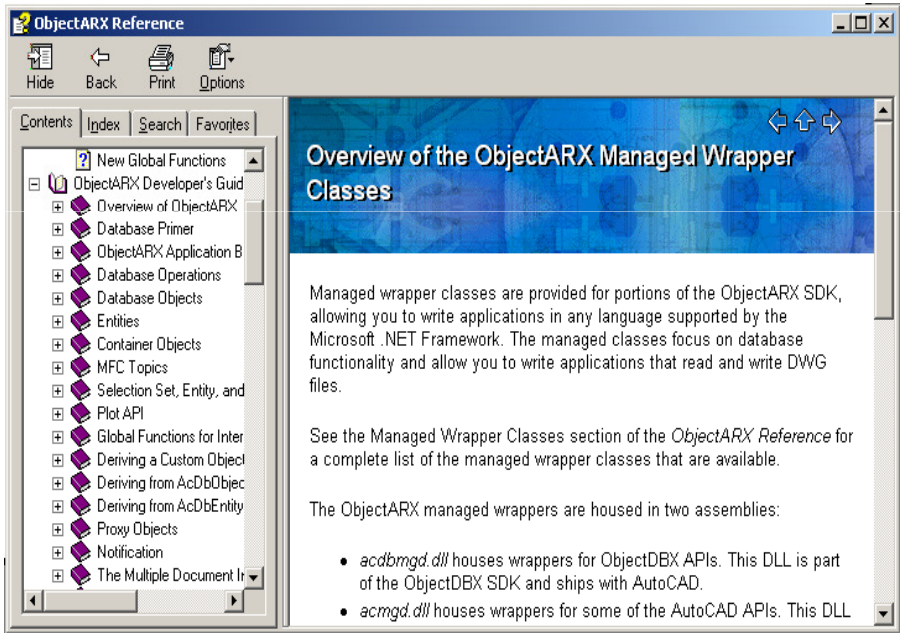
- Overview of .NET.
- **AutoCAD .NET Visual Studio project settings – Hello World!**
- User Interaction - Simple User Input and Entity Selection
- Database Fundamentals – Symbol tables, Transactions
- Database Fundamentals – Dictionaries, XRecords, Table Traversal
- More User Interaction – Advanced Prompts
- User Interface design - WinForm Dialogs and Palettes
- Event handling – Reacting to AutoCAD Events in .NET.

AutoCAD .NET API Documentation

How do I get started?

ObjectARX SDK Includes:

- SDK Samples!
- ObjectARX Developer's Guide
- Managed Reference Guide
  - Acad_Mgd.chm



ADN website

- DevNotes

Development Environment

Microsoft Visual Studio 2002(7.0), 2003 (7.1) or 2005

AutoCAD 2007

Microsoft Windows 2000

- Service pack 2

or

Microsoft Windows XP

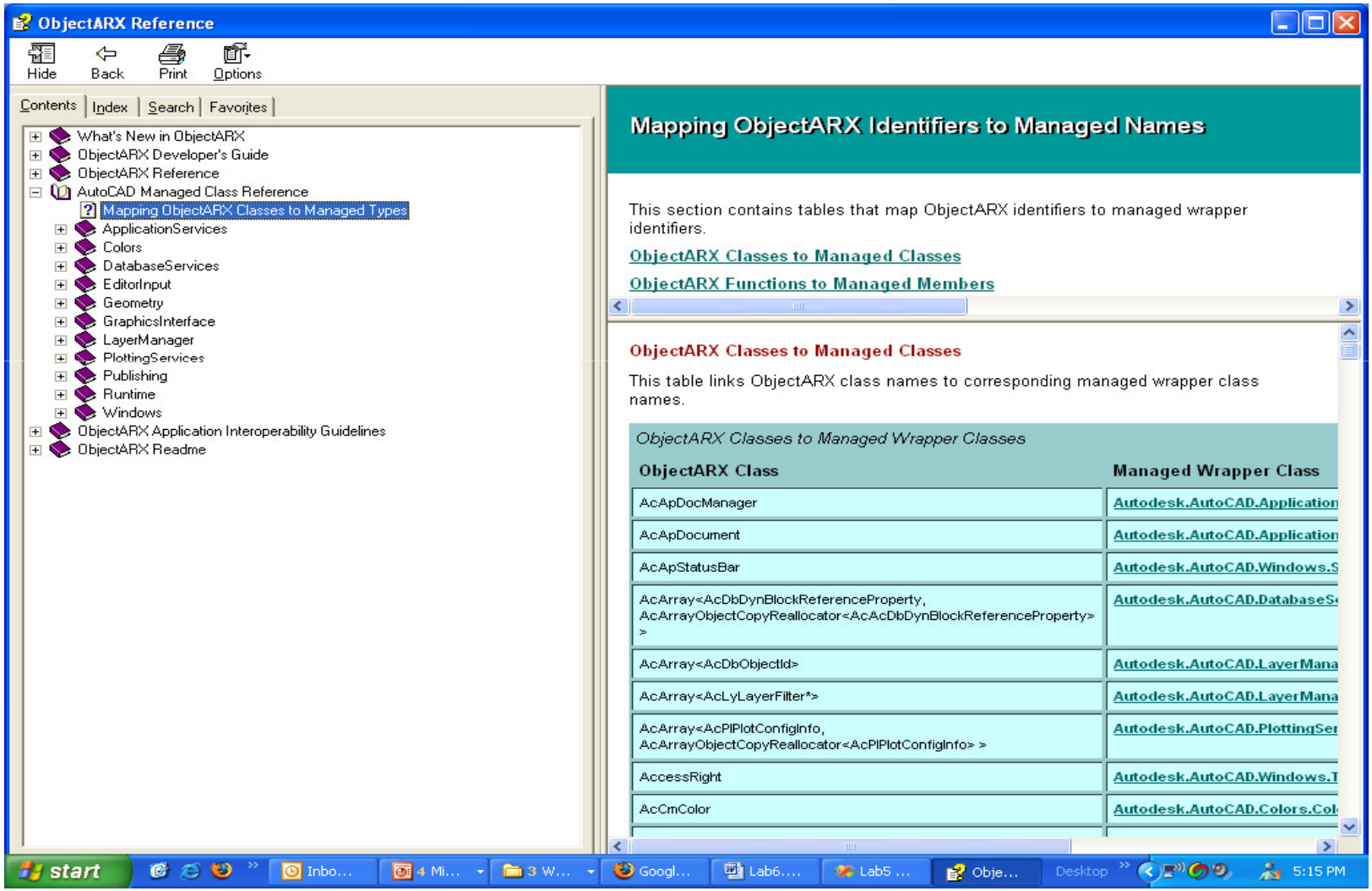
.NET Debugging Tools

- Reflector
  - Browse .NET assemblies, disassemble, decompile
    - http://sharptoolbox.madgeek.com
- Ildasm
  - Disassemble .NET assemblies
    - Visual Studio Tools
- Fuslogv
  - Diagnose load time problems
    - Visual Studio Tools
- FxCop
  - Check conformance with Design Guidelines
    - http://www.gotdotnet.com/team/fxcop/

Snoop Tools (for AutoCAD's database)

- ArxDbg (C++) ObjectARX SDK
- MgdDbg(C#) ADN

ObjectARX – Managed Map



Visual Studio project settings– Hello World!

- Start with a Class Library application type with DLL output.
- Add references to AutoCAD's managed assemblies
  - acdbmgd.dll
    - Database services and DWG file manipulation (like ObjectDBX)
  - acmgd.dll
    - AutoCAD Application specific (like ObjectARX)
  - Find them in the AutoCAD install folder
    C:\Program Files\AutoCAD 2007\

Visual Studio project settings– Hello World!

Reference namespaces you will use in your project

In VB.NET Use Imports keyword:

Imports Autodesk.AutoCAD.ApplicationServices

Access to the AutoCAD application

Imports Autodesk.AutoCAD.EditorInput

Access to the AutoCAD editor

Imports Autodesk.AutoCAD.Runtime

Command registration

Visual Studio project settings– Hello World!

Add a simple command – HelloWorld

- Make a function an AutoCAD command by adding an *attribute*

```
Public Class Class1
    <CommandMethod("HelloWorld")> _
    Public Function HelloWorld()
    End Function
End Class
```

  - The attribute is added to the metadata for that function

Visual Studio project settings– Hello World!

To print a string to command line

- Get the editor *object* for the active document

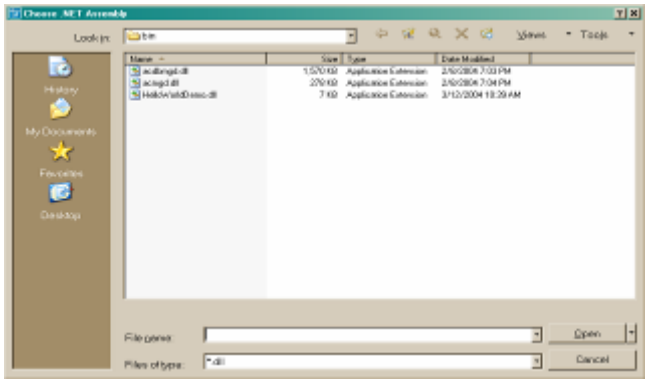
```
Dim ed As Editor =
Application.DocumentManager.MdiActiveDocument.Editor
```

- Call the editor's WriteMessage method

```
Public Class Class1
    <CommandMethod("HelloWorld")> _
    Public Function HelloWorld()
```

Loading .NET assembly

- NETLOAD command
- Demand Load (Registry)
  - Startup
  - On command invocation
  - On request
    - **From another application**
  - On proxy detection



[HKEY_LOCAL_MACHINE\SOFTWARE\Autodesk\AutoCAD\R17.0\ACAD-5001:409\Applications\AcLayer]
"DESCRIPTION"="AutoCAD Layer Manager"
"LOADER"="C:\\Program Files\\AutoCAD 2007\\aclayer.dll"
"LOADCTRLS"=dword:0000000e
"MANAGED"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\Autodesk\AutoCAD\R17.0\ACAD-5001:409\Applications\AcLayer\Commands]
"LAYER"="LAYER"

[HKEY_LOCAL_MACHINE\SOFTWARE\Autodesk\AutoCAD\R17.0\ACAD-5001:409\Applications\AcLayer\Groups]
"ACLAYER_CMDS"="ACLAYER_CMDS“

Use Installers to set these keys!

Lab 1 – Hello World!

Class Agenda

Lectures and Labs

- Overview of .NET.
- AutoCAD .NET Visual Studio project settings – Hello World!
- **User Interaction - Simple User Input and Entity Selection**
- Database Fundamentals – Symbol tables, Transactions
- Database Fundamentals – Dictionaries, XRecords, Table Traversal
- More User Interaction – Advanced Prompts
- User Interface design - WinForm Dialogs and Palettes
- Event handling – Reacting to AutoCAD Events in .NET.

Prompting for User Input

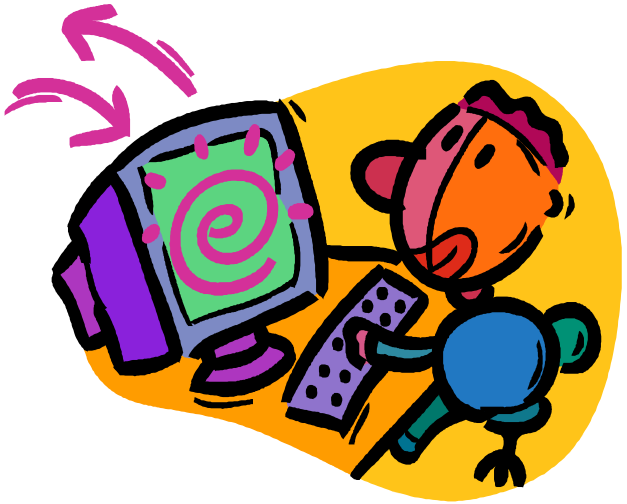
- Use Prompt*XXX*Options to set the parameters for prompting
  - *XXX* is the value type we want to prompt, such as Angle, String, Distance, Corner etc.
  - Use Message and Keywords properties to set the prompt string and list of keywords
  - Use Allow*YYY* to set conditions for prompting. *For e.g., AllowNegative*
- To prompt, use Editor's GetXXX functions
  - Examples - GetAngle, GetString, GetDistance, GetCorner etc
  - Pass Prompt*XXX*Options into Get*XXX*
- Result of prompting stored in PromptResult or derived types

ObjectARX 2007 Wizards

AppWizard – Templates for a VB.NET or C# application

Add-In

Lab 2 – Simple User Input



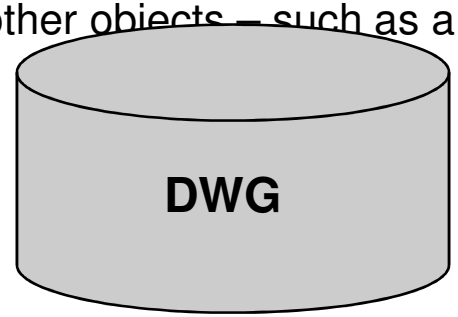
Class Agenda

Lectures and Labs

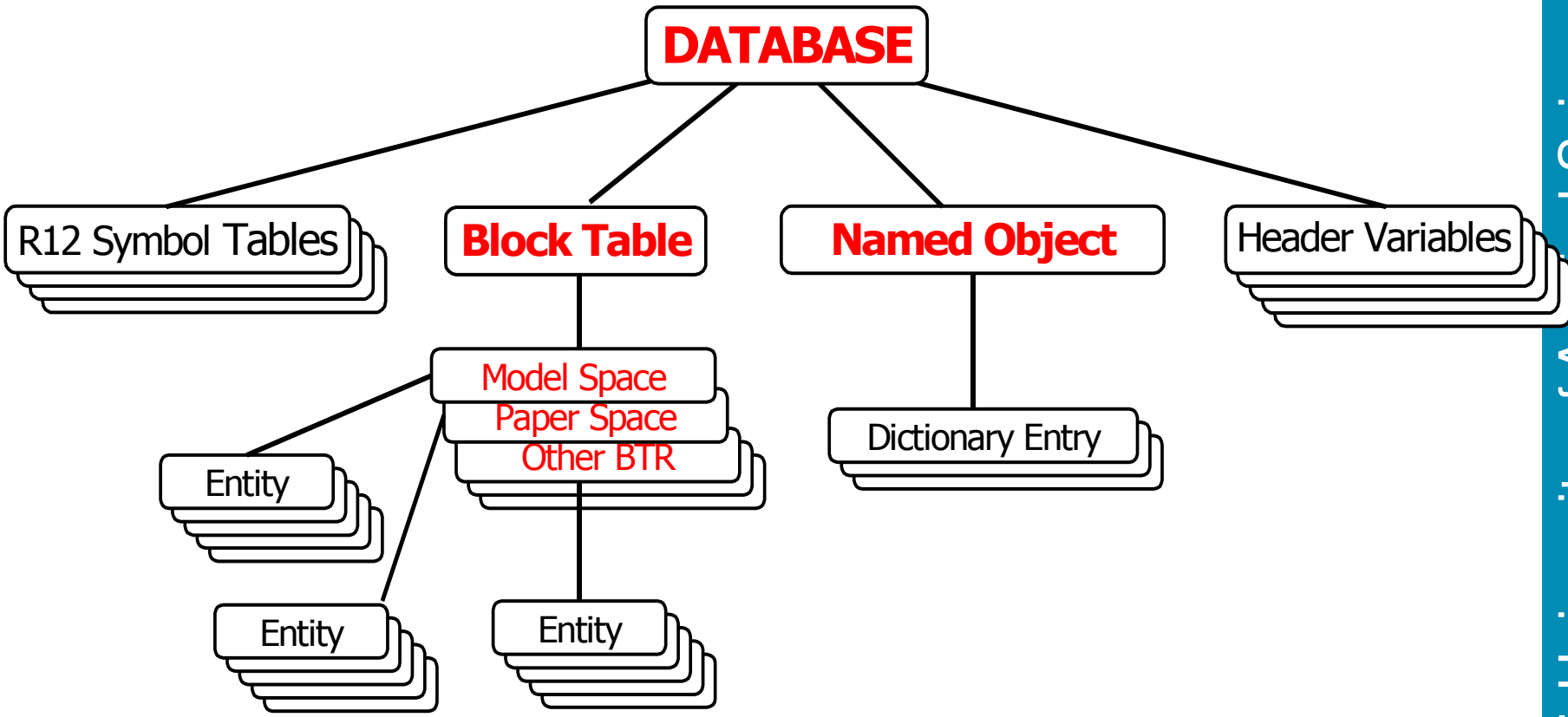
- Overview of .NET.
- AutoCAD .NET Visual Studio project settings – Hello World!
- User Interaction - Simple User Input and Entity Selection
- **Database Fundamentals – Symbol tables, Transactions**
- Database Fundamentals – Dictionaries, XRecords, Table Traversal
- More User Interaction – Advanced Prompts
- User Interface design - WinForm Dialogs and Palettes
- Event handling – Reacting to AutoCAD Events in .NET.

AutoCAD Drawing Database

- In-Memory representation of the Dwg File
  - Objects are stored hierarchically in the database - Db Structure
  - All objects have identities – ObjectId, like *Primary Key* in a relational database
  - Objects are always accessed in a transaction
    - The transaction defines the boundary of database operations
    - Objects have to be opened first in a transaction before they can be used
  - Objects can refer to other objects – such as a line having a reference to a layer



Database Structure



Database Components

- Symbol Tables
  - Examples Layer Table, Linetype Table, Textstyle Table etc.
  - Containers to store Symbol Table *Records*
    - Example LayerTable*Record*, LinetypeTable*Record* etc
  - All Symbol Tables have common methods of a container such as
    - Add – to add a record
    - Item – to lookup an entry with a search string
    - Has – To know if an entry exists
  - Is enumerable
  - Each symbol table can hold only records of a specific type
    - For example, a LayerTable can hold only LayerTableRecords

Database Components

- Block Table
  - Child of Symbol Table
    - Holds a collection of BlockTableRecords (or Block Definitions)
      - BlockTableRecords (BTR) in turn is a container that holds only graphical entities (i.e., those derived from Entity type)
      - Two default BlockTableRecords cannot be removed – Model Space and one Paper Space
  - Searchable with a string or an ObjectId (more on this later)
  - Neither the Block Table nor the Symbol Tables are created by the user. They are created by default when you create a new database.

- Create a block definition and use ArxDbg or MgdDbg to see where the block definition is stored and peek into its contents

Getting a Database Object

- Construct One
  - In Memory

- Get the one currently active in AutoCAD
  - HostApplicationServices.WorkingDatabase()

Object Identity - ObjectID

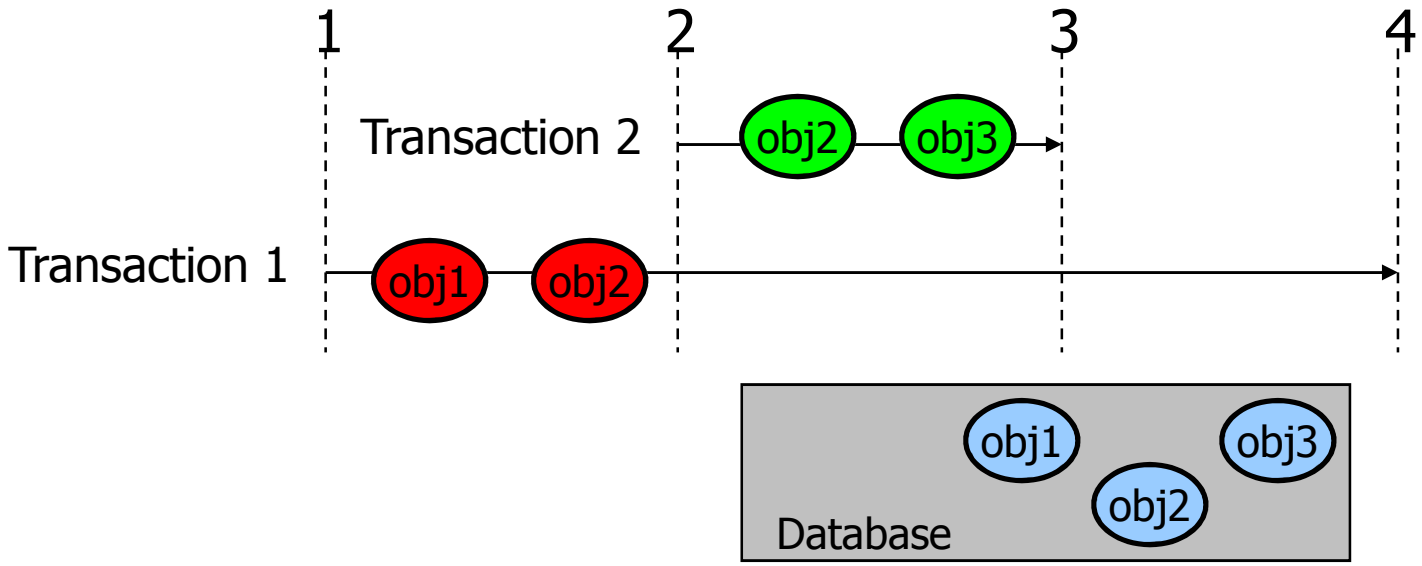
- All Objects that exist in the database have an ObjectId
  - Is *unique* per session (or instance) of AutoCAD
  - Is generated automatically for an object when it is added to the database
  - Non-database resident objects do not have an ObjectId set
  - Can be cached to open an object later

- Get it using ObjectId property

Transactions

- Transactions
  - Sets the boundary for database operations
  - Handles exception cleanly
  - Operates with a single Undo filer
  - Can be
    - committed – All database operations are saved
    - rolled back – All database operations are aborted
  - Can be nested

Nesting Transactions



1. Client starts Trans1 and gets Obj1 & Obj2
2. Client starts Trans2 and gets Obj2 & Obj3
3. Client commits Trans2
- **Trans2 changes are committed**

4a. Client commits Trans1
- **Trans1 changes are committed**

4b. Client aborts Trans1 instead
- **Trans1 (and Trans2) changes are rolled back**

Transactions

- To start a transaction, use Databases' StartTransaction()

  For example:

```
Dim db As Database =
HostApplicationServices.WorkingDatabase() 'Get the current db
used by AutoCAD

Dim trans As Transaction =
db.TransactionManager.StartTransaction() ' begin the transaction
```

- To commit a transaction, use transaction's Commit()
- To Abort a transaction, use transaction's Abort()

Transactions

- Standard db access with Transactions

```
Public Function MyFunc()

    'Get the database current in AutoCAD
    Dim db As Database = HostApplicationServices.WorkingDatabase()

    'Start a transaction using the database transaction manager
    Dim trans As Transaction = db.TransactionManager.StartTransaction()

    Try
      'Do all database operations here
      'Lets get the block table from the database
       'Drill into the database and obtain a reference to the BlockTable
      Dim bt As BlockTable = trans.GetObject(db.BlockTableId, OpenMode.ForWrite)
      'Everything successful, so commit the transaction
      trans.commit()
    Catch
      trans.Abort()
    Finally
      'All ok. Call Dispose explicitly before exiting
      trans.dispose()
    End Try

End Function
```

Transaction - Opening an Object

- **Use transaction's GetObject to open an object**
  - The first parameter is the ObjectId
  - Second parameter is the open mode
    - ForRead – Access but not Modify
    - ForWrite – Modify and Access
    - ForNotify – When the object is notifying

```
Dim bt As BlockTable = trans.GetObject( _
                                db.BlockTableId, _
                                OpenMode.ForWrite _
                                )
```

Adding an Object to the database

- Find the right owner to add a newly created object to
  - All objects have exactly *one* owner
    - The owner can be the database itself!
  - For example, newly created LayerTableRecord can *only* be added to the Layer Table, its owner, or a newly created entity can be *only* added to a block table record

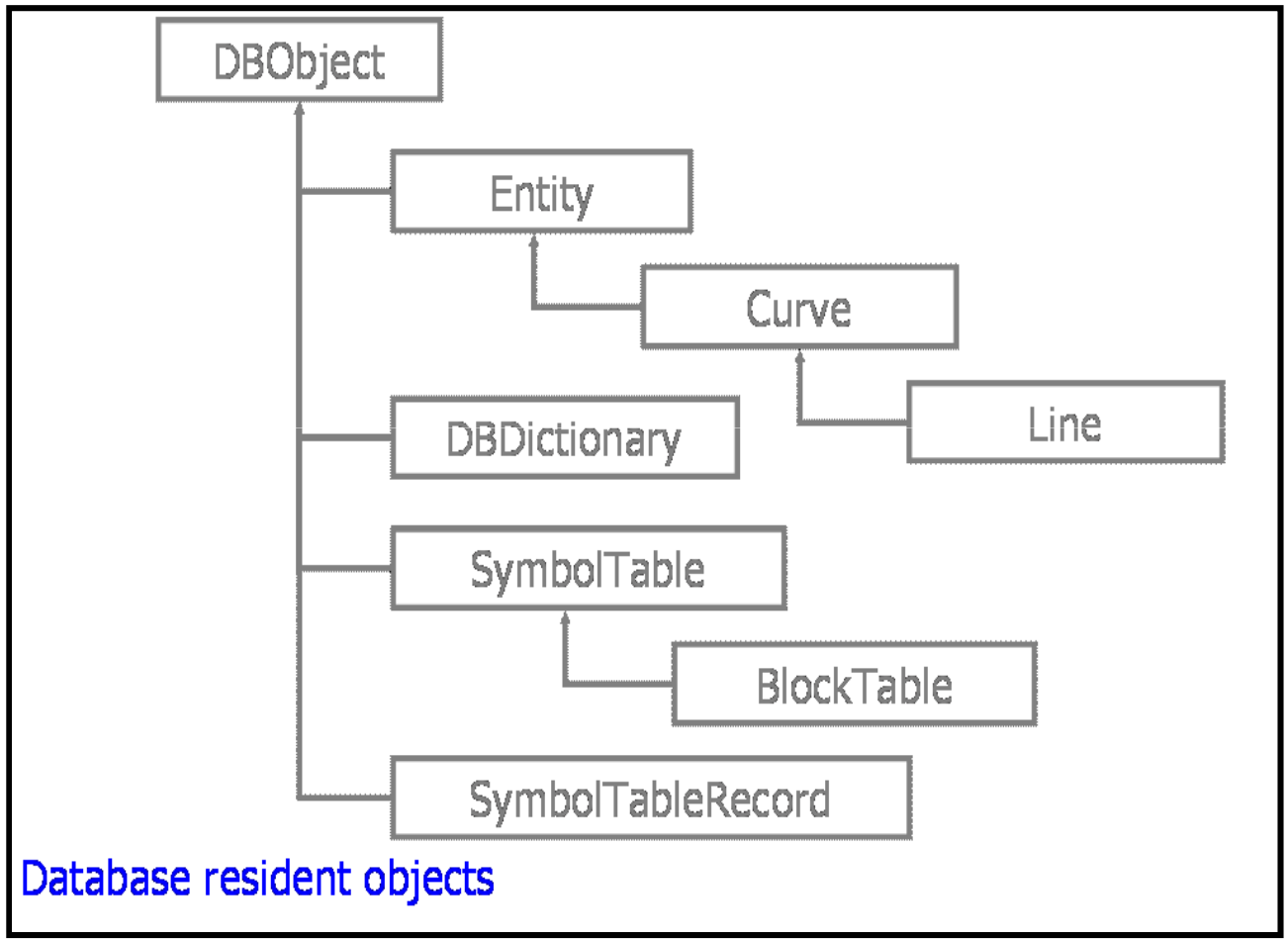
- Use Add method for adding Symbol Table Records to add to Symbol Table

- Use AppendXXX to add add other kinds of objects to its owners For example
  - AppendEntity to add to BlockTableRecord

- Once an object is added to an owner, always let the transaction know! For example

```
newBtr.AppendEntity(circle) 'Add our circle to its owner the BTR
trans.AddNewlyCreatedDBObject(circle, True)
```

Important Managed Classes

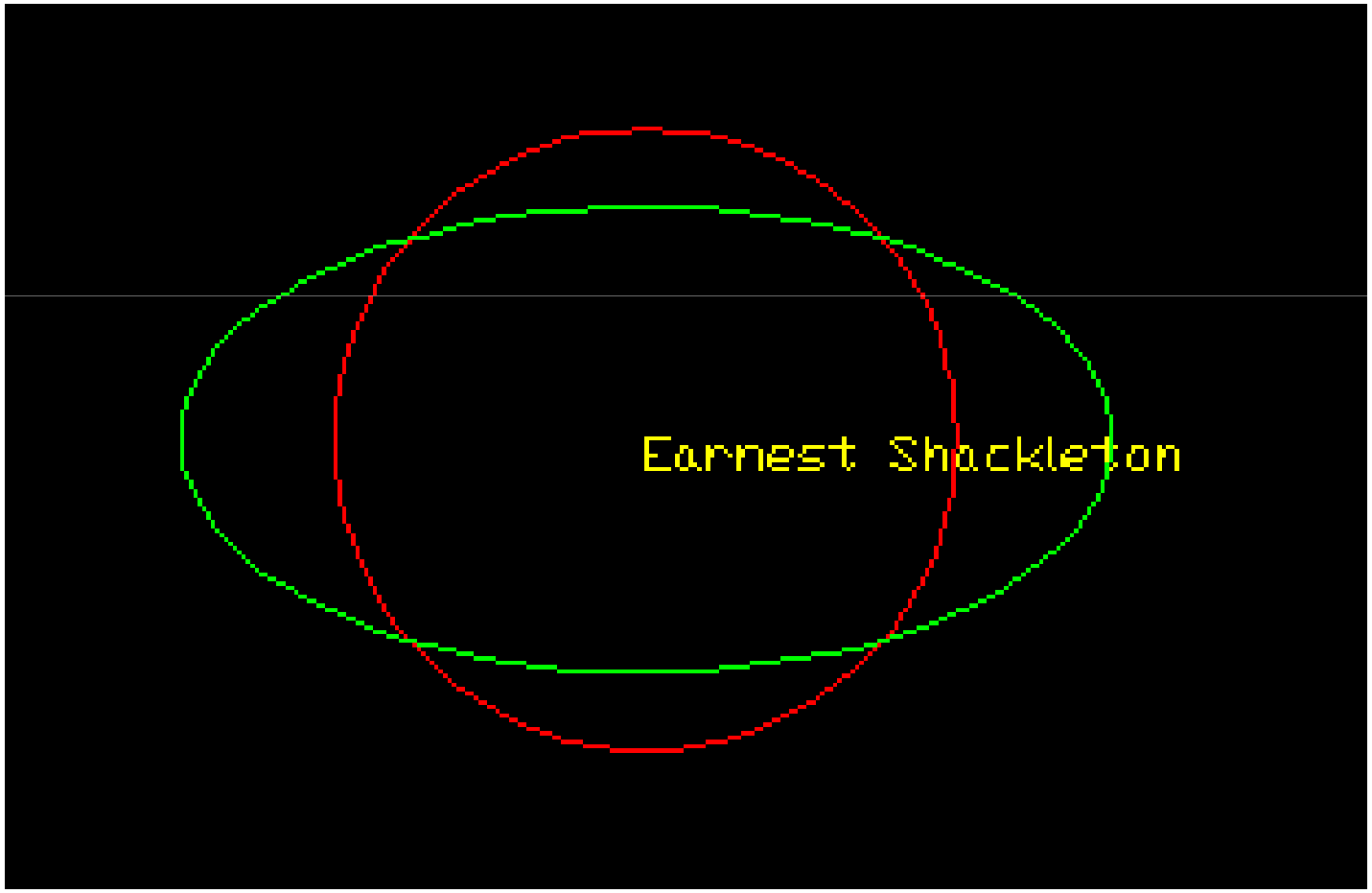


Object memory management

- Note managed objects *wrap* an unmanaged C++ object!
- So we create them with New, Do they need to be disposed?
  - No - garbage collection disposes the object when it wants to reclaim memory
    - If the object is not in the database — this *deletes* the underlying unmanaged object
    - If the object is in the database – this *Closes* the underlying unmanaged object
- If opened in a transaction, disposing or committing the transaction closes it automatically! – Clean memory management.

Lab 3

Create Employee block Definition



Class Agenda

Lectures and Labs

- Overview of .NET.
- AutoCAD .NET Visual Studio project settings – Hello World!
- User Interaction - Simple User Input and Entity Selection
- Database Fundamentals – Symbol tables, Transactions
- **Database Fundamentals – Dictionaries, XRecords, Table Traversal**
- More User Interaction – Advanced Prompts
- User Interface design - WinForm Dialogs and Palettes
- Event handling – Reacting to AutoCAD Events in .NET.

Recommended Transaction Use

- Standard db access with Transactions

```
Public Function MyFunc()

    'Get the database current in AutoCAD
    Dim db As Database = HostApplicationServices.WorkingDatabase()

    'Start a transaction using the database transaction manager
    Using trans As Transaction = db.TransactionManager.StartTransaction()

      'Do all database operations here
      'Lets get the block table from the database
       'Drill into the database and obtain a reference to the BlockTable
      Dim bt As BlockTable = trans.GetObject(db.BlockTableId, OpenMode.ForWrite)

      'Everything successful, so commit the transaction
```

Dictionaries and XRecords

- Dictionaries (Type DbDictionary)
  - Containers to hold data only
  - Holds other Dictionaries
  - Holds non-graphical Objects (derived from DbObject but not DbEntity!)
  - Is enumerable
    - Each item has a string key
    - Items searchable with a string key using GetAt() or Item
- Two *Root* dictionaries
  - Named Objects Dictionary (NOD)
    - Owned by the database
    - Available by default
    - Used to store database level data
  - Extension Dictionary
    - Owned by an Entity
    - Created by the user only when needed
    - Used to store entity level data
- Use SnoopDb to look where the dictionaries are stored

Dictionaries and XRecords

- XRecord
  - Data containers
  - Holds data in a Resbuf chain (Result Buffer)
    - Resbuf – Linked List of TypedValues (DataType–Value pair)
  - No “Key” to search values within a Resbuf. Should know the order data is stored in the list
  - XRecords can be added to Dictionaries
    - If stored in NOD –database level data
    - If stored in Extension Dictionary – entity-level data

Get NOD

- To *get* the NOD for the database

```
Dim db = HostApplicationServices.WorkingDatabase

Dim NOD As DBDictionary =
trans.GetObject(db.NamedObjectsDictionaryId, OpenMode.ForWrite, False)
```

- To *create* an ExtensionDictionary for an entity

```
myEntity.CreateExtensionDictionary()
```

Iterating Through Containers

Objects that are enumerable

- Symbol Tables
- Block Table Records
- Dictionaries
- Polylines
- PolyFaceMesh & PolygonMesh
- ACIS Solids
  - Called traversers
- BlockReferences (Inserts)
  - Only useful when attributes are present

Symbol Table Traversal

- Start with a database pointer
  - HostApplicationServices.WorkingDatabase – Get used to this one!

- Iterate down into each sub-table from there…

```
Dim db As Database = HostApplicationServices.WorkingDatabase()

Dim bt As BlockTable = trans.GetObject(db.BlockTableId,
OpenMode.ForWrite)

Dim id As ObjectId
For Each id In bt
    Dim btr As BlockTableRecord = trans.GetObject(id,
```

Runtime Type Identification (RTTI)

- Type information (such as constructors, methods, properties, events etc) are contained in the Metadata

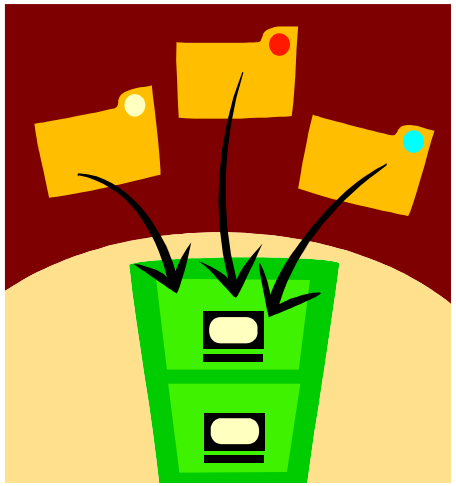
- Query metadata at runtime using types defined in System.Reflection namespace
  - Not only to query type information but also dynamically create them.
  - Heart of Reflection is provided by System.Type class
    - The class is abstract. Implementers need to inherit methods to describe the type
    - Get it using
      - Object.GetType
      - TypeOf operator (VB, C# or C++)

Casting*

- VB.NET
  - Casting is automatic by assignment – Only if OptionStrict is turned off
  - Explicit casting – CType(*expression*, *type*)
  - TypeOf – If TypeOf myObject is Line Then…
  - If obj1.GetType() Is obj2.GetType() Then…
- In C#
  - Line myLine = (Line)myObject;
  - 'as' operator - Line myLine = myObject as Line; -
    - if it fails, no exception, but object gets set to null!
  - 'is' operator – if (myObj is Line) …
  - typeof operator – if typeof (obj1) == typeof(obj2)…
  - GetType - If obj1.GetType() == obj2.GetType() …

Lab 4

Adding Custom Data



Class Agenda

Lectures and Labs

- Overview of .NET.
- AutoCAD .NET Visual Studio project settings – Hello World!
- User Interaction - Simple User Input and Entity Selection
- Database Fundamentals – Symbol tables, Transactions
- Database Fundamentals – Dictionaries, XRecords, Table Traversal
- **More User Interaction – Advanced Prompts**
- User Interface design - WinForm Dialogs and Palettes
- Event handling – Reacting to AutoCAD Events in .NET.

More User Interaction

- PromptsXXXOptions is used to control prompting such as
  - Set the *Message*

Enter Number of Sides:

  - Set *Keywords*

Enter Number of Sides [Triangle/Square/Pentagon] :

  - Set *Defaults*

Enter Number of Sides [Triangle/Square/Pentagon] <3>:

  - Set *Allowed* values

Enter Number of Sides [Triangle/Square/Pentagon] <3>: -5

Value must be positive and nonzero.

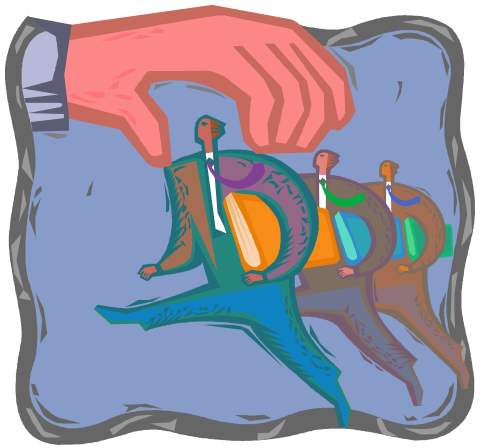
Prompts

Types:

- PromptPointOptions
- PromptStringOptions
- PromptDoubleOptions
- PromptAngleOptions
- PromptCornerOptions
- PromptDistanceOptions
- PromptEntityOptions
- PromptIntegerOptions
- PromptKeywordOptions
- PromptNestedEntityOptions
- PromptNestedEntityOptions
- PromptSelectionOptions
- Etc.

Lab 5

Prompts and Selection



Class Agenda

Lectures and Labs

- Overview of .NET.
- AutoCAD .NET Visual Studio project settings – Hello World!
- User Interaction - Simple User Input and Entity Selection
- Database Fundamentals – Symbol tables, Transactions
- Database Fundamentals – Dictionaries, XRecords, Table Traversal
- More User Interaction – Advanced Prompts
- **User Interface design - WinForm Dialogs and Palettes**
- Event handling – Reacting to AutoCAD Events in .NET.

User Interface Design

- AutoCAD Defined
  - Menus – Application level menu, Context menu
  - Dialogs
    - AutoCAD's Enhanced Secondary Windows (Palettes)
    - Color, Linetype, Lineweight, OpenFile dialogs
  - Tabbed Dialog Extensions (to Options Dialog)
  - Status Bar
  - Tray
  - Drag-Drop
  - And more. Explore Autodesk.AutoCAD.Windows namespace

- Windows Defined
  - Windows Forms (Winform)
  - Host of other controls defined in CLR

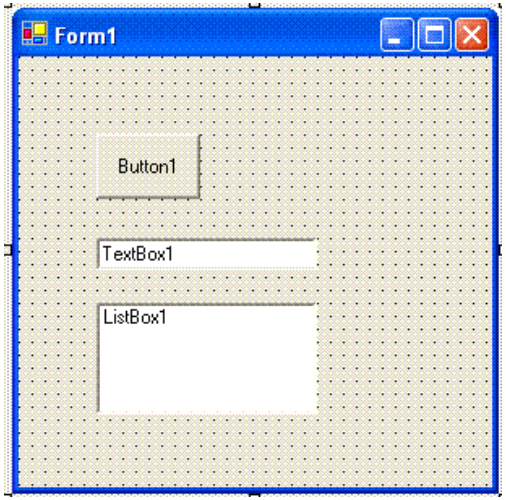
User Interface

- Add Context Menu
  - Application Level – Application.AddDefaultContextMenuExtension
  - Object Level – Application. AddObjectContextMenuExtension –per RXClass

- Showing Modal and ModelessDialogs
  - Application.ShowModalDialog
  - Application.ShowModelessDialog
  - Persists size and position automatically

Lab 6

User Interface Design



Class Agenda

Lectures and Labs

- Overview of .NET.
- AutoCAD .NET Visual Studio project settings – Hello World!
- User Interaction - Simple User Input and Entity Selection
- Database Fundamentals – Symbol tables, Transactions
- Database Fundamentals – Dictionaries, XRecords, Table Traversal
- More User Interaction – Advanced Prompts
- User Interface design - WinForm Dialogs and Palettes
- **Event handling – Reacting to AutoCAD Events in .NET.**

Handling Events

- Event
  - message sent by an object to notify something has happened
  - message is received in a function call by one or more listeners
  - event sender only requires function pointer to dispatch a message
  - any interested party can implement the function and receive the event
  - function must have a specific signature that the sender requires
  - Use .NET delegates to '*wire*' sender and receiver

- Delegates
  - Like a class (can be instantiated) but with a signature
  - Holds references to functions having same signature
  - Like 'Type-Safe' function pointer
  - Can encapsulate any method which matches the specific signature

Using Delegates

- Delegates in AutoCAD's .NET API usually have 'EventHandler' suffix
  - Lookup the signature of the delegate in the object browser
- Implement a function with same signature
- Instantiate the delegate passing address of the function into its constructor
- Add the delegate instance to sender's list of listeners
  - C#, use += operator
  - VB, use AddHandler

```
Delegate myDelegate = new Delegate(address of myFunction);
EventSender.Event += myDelegate;

myFunction(delegate signature)
{
}
```

- Don't forget to remove the listener!

Event Handling - Example

- Create the event handler (callback)

```
Sub objAppended(ByVal o As Object, ByVal e As ObjectEventArgs)
    MessageBox.Show("ObjectAppended!")
    'Do something here
    'Do something else, etc.
End Sub
```

- Associate the event handler with an event

```
Dim db As Database
db = HostApplicationServices.WorkingDatabase()
AddHandler db.ObjectAppended, NewObjectEventHandler(AddressOf
objAppended)
```

- Disconnect the event handler

```
RemoveHandler db.ObjectAppended, AddressOf objAppended
```

Lab 7

Event Handling

Programming Tools / ObjectARX

- **ObjectARX**
  - The ObjectARX® programming environment provides object-oriented C++, C# and VB .NET application programming interfaces for developers to use, customize, and extend AutoCAD software

Programming Tools / ObjectARX & .NET

- .NET Customize and extend AutoCAD and AutoCAD-based products with direct access to AutoCAD database structures, native command definition and more, using any .NET supporting language.

Programming Tools / ActiveX, VBA

- **ActiveX (COM Automation)**
- **Using the ActiveX® (COM Automation) interface in AutoCAD software, you can build applications with a variety of programming technologies, including Microsoft® Visual C++®, Microsoft® Visual Basic® for Applications (VBA), and Microsoft® Visual Basic® (VB).**
- **Microsoft Visual Basic for Applications The combination of the powerful ActiveX® Automation object model in AutoCAD® and Microsoft® Visual Basic® for Applications (VBA) presents a compelling framework for customizing the AutoCAD software program.**
- **Visual LISP Customize AutoCAD as well as its runtime**

Esan lisäykset

What's New AutoCAD2007

- ***New Application Programming Interfaces (APIs)***
  - Microsoft .NET Framework Programming Support
- **API Compatibility**
  - Autodesk does not support binary compatibility for AutoCAD® 2007. ObjectARX applications and Object Enablers built for prior releases of AutoCAD will need to be rebuilt to work with AutoCAD 2007.
- **Platform Changes with AutoCAD 2007**
  - ObjectARX applications need to be built with Microsoft® Visual Studio® .NET 2005 (Visual C++ 2005, version 8.0).
  - Note: Visual Studio® .NET 2003 is not supported.

What's new in AutoCAD2008

- ***New Application Programming Interfaces (APIs)***
- **Microsoft .NET Framework Programming Support**
  - The .NET API includes facilities to implement functions in any .NET language that can be called from VisualLISP. .NET developers can easily modify menus, toolbars and other UI elements using the CUI API.
- **API Compatibility**
  - AutoCAD® 2008 supports applications built for AutoCAD® 2007.
  - ObjectARX applications and Object Enablers built for AutoCAD 2006 or earlier versions will need to be rebuilt to work with AutoCAD 2008.