

# *Automating with ID Works SDK*

Leslie Dreyer Kalra  
Kutztown University  
NACCU 2010

# About our program

- State university in southeastern Pennsylvania, midway between Allentown and Reading
- ~10,000 students, 60% commuters
- Primarily known for Education programs and Communication Design

# What is an SDK?

- SDK = Software Developer's Kit
- Provides a software environment for a developer and/or...
- Provides hooks into a software package that let a programmer enhance it

# Why use ID Works SDK?

- ID Works doesn't quite do what we need it to do.
- We want to automate some things ID Works does manually.
- We have an unusual setup that ID Works doesn't handle.
- We want to walk staff through a procedure.
- We like to play with software toys.

# KU and our challenge

- Kutztown University has ~10,000 students.
- In May 2009, KU converted student ID numbers from SSNs to unique IDs.
- Administration decided to print the new ID numbers on the cards. Therefore...
- All students needed to get new cards.
- We chose not to change the card number if old card was turned in.

# What KU used the SDK for

- Student recard:
  - Speed!
  - Reduction of human error
  - Automatic logging
- Freshman Orientation
  - Batch print all IDs printed for that day's event.
  - Speed!
- Prompt staff through carding procedures

# Other uses

- Replace all or part of the ID Works interface
- Use a capture source that ID Works doesn't know about natively (requires purchase of Capture Server for each station)
- Use a printer setup that ID Works doesn't know about natively (requires purchase of Print Server for each station)

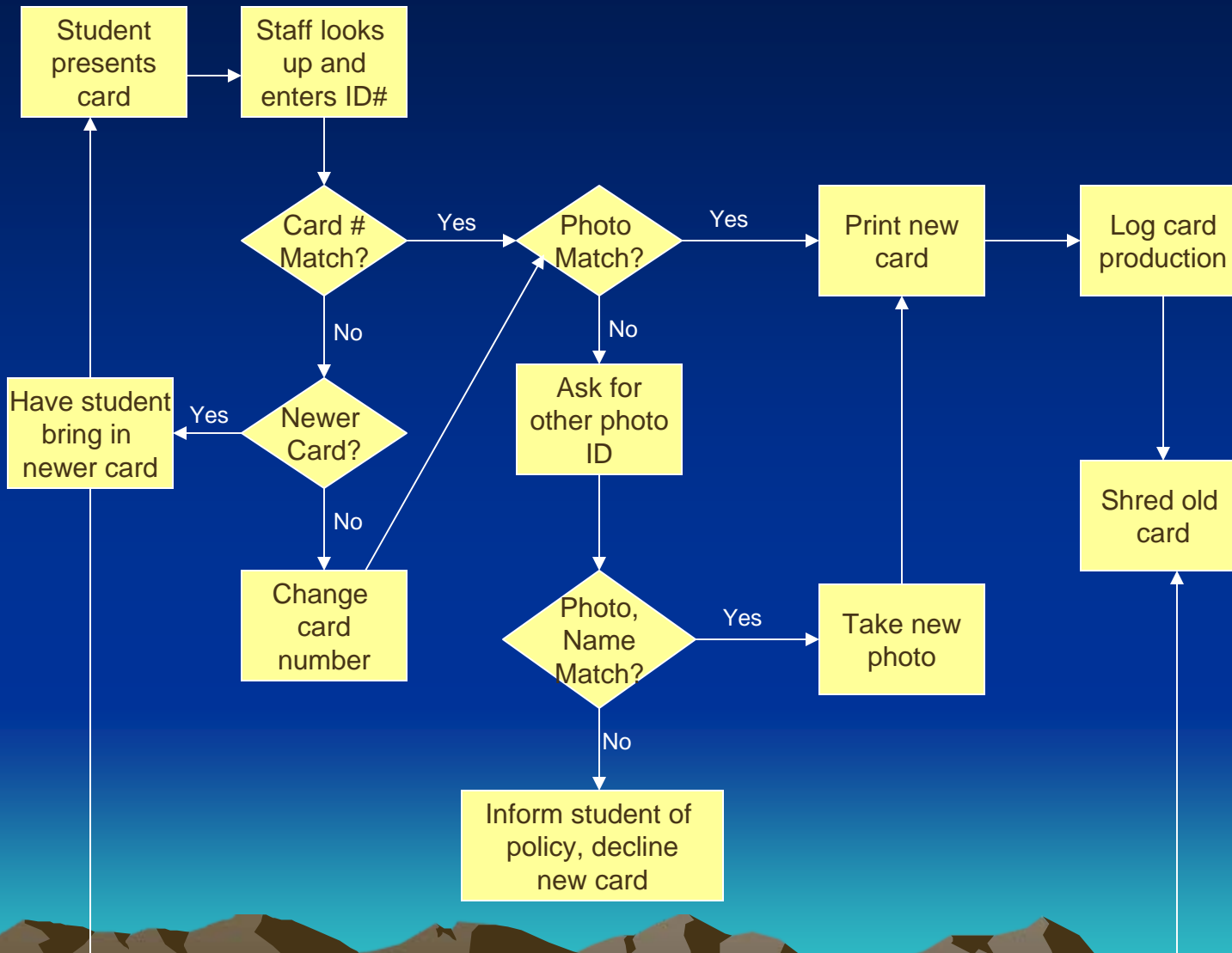
# What the SDK includes

- Datacard Workstation Utility
- **IDWRegtool**
- RegisterInterops.cmd
- **Programmer's Reference Guide**
- **Samples in VB.NET and C++**

# Event Handlers: A different programming paradigm

- **Simple procedural programming:** *start at the beginning, go to the end, then stop*
- **Event-driven programming:** *set up the environment, wait for something to happen, then react.*
- **Getting started:** *Flowchart process, identify what can be automated.*

# Recard Production Flow



# Object-Oriented Programming

- Objects belong to classes, classes can have hierarchy
- Properties describe objects
- Methods act on objects
- Properties and methods can be inherited and overloaded
- Before a class can be used, an *instance* of it must be created. It refers to itself as *Me* (VB.NET) or *this* (C++, C#)

# Example: Ball Class

Properties

Methods

# Event Handler as Mom

- Whenever an event takes place, ID Works checks in with the Event Handler
- Event Handler can allow/disallow, cause other things to happen before/after, require things to happen before continuing.
- Just like Mom!

# ID Works events and methods

- An event can be a mouse click, keystroke, project open, record deletion, update, etc.
- When ID Works does most things, there is an event before it happens (`_PRE`) and another one afterwards (`_POST`).
- ID Works methods make things happen programmatically instead of interactively.
- Methods can fire events, too, so watch for side effects.

# m\_processing

- A method called because of an event can cause other events. This can lead to unexpected behavior.
- `M_processing` is set to `true` when an event fires. The last thing done in the `select` statement is to set it to `false` again.
- Setting this to `true` prevents other events from triggering.

# IDataEntry3 object

## Some events we used

- EV\_SEARCH\_PRE
- EV\_READRECORD\_POST
- EV\_UPDATE\_POST
- EV\_PRINT\_POST
- EV\_PROJECT\_UNLOAD

## Some methods we used

- ExecuteSQLSearch()
- GetFieldData()
- ClearForm()
- ClearRecordSet()
- Capture()
- UpdateRecord()
- PrintCard()

# The VB.Net Event Handler template

- Run IDWRegtool to install templates
- Start new ID Software->VB Event Handler project in Visual Studio
- Set up debugging in Visual Studio with Projects->Properties->Debug
- EventNotify() sub in EventHandler.vb is Mom
- m\_ide3 refers to your ID Works session object

# Try..Catch..Finally

- Try: do stuff
- Catch: deal with problems
- Finally: clean up
  
- Variables in try block not available in catch block
- Good programming practice in general

# Using your event handler

- In ID Works Designer, open your project
- In File->Project Properties, choose the Event Handler tab
- Add event handlers from unassigned list to assigned list, in order
- Back out event handlers by moving them back to the unassigned list
- DEMO

# Debugging

- In Visual Studio, open Project Properties
- On *Debug* tab, click on *Start External Program*
- In the adjacent box, enter path to the ID Works program *DataEntry.exe*
- Click on the green arrow to fire 'er up!

# Deploying

- Compile your code in Visual Studio
- Copy IDWRegtool to ID station
- Copy .dll file to ID station
- Run IDWRegtool on ID station to register your new event handler

# Card Reader Config

- Need only some data on the card: ID number, name, card number
- Configured to read Tracks 1 and 2 into this format:

9999999999^DENT/STU A^6220010000999999=

- Event handler expects this format

# Sharing our efforts

- Current: ad-hoc sharing on the listserv
- Proposed: a browsable location to which we can ftp the programs we've found useful (NACCU? Sourceforge?)
- Caveat: be sure we have rights to share the code we provide
- Usual software caveats (e.g. no warranty) apply
- Ideas?!

# Where to find me

Leslie Dreyer Kalra  
Manager, One Card Office  
610-683-4829  
dreyer@kutztown.edu