

VITAL INFORMATION

Subject(s):	Careers, Computer Fundamentals 1-2
Topic or Unit of Study:	Integrated Unit
Grade/Level:	9-12
Objective:	<p>At the conclusion of this lesson students will be able to:</p> <ol style="list-style-type: none">1. Add a ComboBox to another control (Form or TabPage).2. Name the ComboBox and set simple properties including DropDownStyle.3. Write code to populate the control at runtime.4. Program the application to react to mouse click events by extracting information from the currently selected ComboBox item and using that information appropriately.

Summary:	<p>Students add and configure one or more ComboBoxes for use in their Computer Integration project. ComboBoxes display lists of options to the user and allow for the selection of one or more, possibly an option not even contained in the list, depending on the properties the programmer has enabled. In order to store two pieces of information for each option in the list, likely a URL and a Name or Title associated with it, students make use of Classes to encapsulate the data.</p>
-----------------	--

IMPLEMENTATION

Learning Context:	<p>Students have completed code for the Schedule tab of their application and are now adding the controls which will allow them to present their Computer Integration findings. Depending on their plans, they may add controls in any order. However, since the ComboBox is dependent on the NamedUrl Class, students should have finished the Class activity before starting this one.</p>
--------------------------	--

Procedure:	<ol style="list-style-type: none">1. The activity is written up in fairly fine detail on a web page which is printed and attached to this lesson plan. It specifies where to find the ComboBox, how to name it, what property is used to control the list style, how to add objects to Items collection, and how to extract information about the currently selected Item when the selection changes. It does not review the big picture, so the stage should be reset using some of the learning context from the introduction to the integrated unit.
-------------------	---

2. Ensure that students can find the web page, ask for questions, and have students start.

3. Since this is a long, sequential unit, more attention needs to be paid to holding it all together and keeping it synchronized. Visit students early and often to ensure that they are keeping up. It may be a good idea to have students demonstrate their progress beginning five minutes before the bell rings.

Differentiated Instruction:

There is little differentiation in instruction, but an expectation that the products are differentiated because of differing courses, schedules, and research findings.

Sample Student Products:

The attached printout of a web page shows an example of how the program might look with one of these controls.

Collaboration:

Students will work individually.

Time Allotment:

1 class period. 55 Min. per class.

Author's Comments & Reflections:

Reflections will follow in a diary entry.

MATERIALS AND RESOURCES

Instructional Materials:

The activity page from the class web site is printed and attached. Students will edit their programs both in Design View and Code View.

Attachments

1. [ComboBox](#)

Resources:

- Technology resources:
Internet Explorer, Visual Basic

STANDARDS & ASSESSMENT

Standards:

 **AZ- Career and Technical Education Programs**

- **Level :** Career Preparation (Grades 10 - 12)
- **Program :** Information Technology CIP No. 15.1200
 - **Option :** Software Development - Option C
 - **Competency :** *3.0 DEVELOP APPROPRIATE WORK HABITS FOR SUCCESSFUL EMPLOYMENT IN INFORMATION TECHNOLOGY
 - **Indicator :** 3.3 Complete tasks accurately
 - **Indicator :** 3.4 Complete tasks with minimal supervision
 - **Competency :** 27.C DEMONSTRATE PROGRAM ANALYSIS AND DESIGN
 - **Indicator :** 27.6c Use stepwise refinement to improve design
 - **Competency :** 28.C USE SOFTWARE TO CREATE PROGRAMS
 - **Indicator :** 28.1c Enter and modify code using a program editor
 - **Competency :** 29.C TEST AND DEBUG TO VERIFY PROGRAM OPERATION
 - **Indicator :** 29.1c Test individual program modules
 - **Competency :** 28.C USE SOFTWARE TO CREATE PROGRAMS

- **Indicator** : 28.2c Compile and execute programs
- **Indicator** : 28.5c Use recognized conventions for naming identifiers and formatting code
- **Competency** : 36.C IDENTIFY WAYS TO INPUT AND OUTPUT INFORMATION
 - **Indicator** : 36.1c Provide user with means to input data on a console and/or GUI
- **Competency** : 27.C DEMONSTRATE PROGRAM ANALYSIS AND DESIGN
 - **Indicator** : 27.4c Determine input and output
- **Competency** : 31.C EMPLOY MODULARITY IN WRITING PROGRAMS
 - **Indicator** : 31.1c Call standard library functions
 - **Indicator** : 31.2c Utilize parameters to pass data into program modules
- **Competency** : 34.C USE SIMPLE DATA TYPES AND STRINGS
 - **Indicator** : 34.4c Write assignment statements for initializing and modifying variables
- **Competency** : 27.C DEMONSTRATE PROGRAM ANALYSIS AND DESIGN
 - **Indicator** : 27.8c Write documentation
- **Competency** : 31.C EMPLOY MODULARITY IN WRITING PROGRAMS
 - **Indicator** : 31.4c Write and use modules that return values
- **Competency** : 34.C USE SIMPLE DATA TYPES AND STRINGS
 - **Indicator** : 34.2c Choose the appropriate data type for a given situation
- **Competency** : 38.C EMPLOY OBJECT-ORIENTED PROGRAMMING TECHNIQUES
 - **Indicator** : 38.4c Write appropriate statements to invoke an object's accessor method(s)
 - **Indicator** : 38.3c Instantiate objects from existing classes

Assessment/Rubrics: By the end of the class period, students should have added, named, configured, and programmed one or more ComboBoxes. Each activity contributes a few criteria to the larger rubric. This activity adds points for name, configuration, and programming both the addition of new items and reaction to click events as described by the ComboBox rubric.

Rubrics

- | |
|--------------------|
| 1. <u>ComboBox</u> |
|--------------------|