Class

Author: Keith Alcock 09/28/2008 06:29:00 PM MDT

VITAL INFORMATION

Subject(s):	Careers, Computer Fundamentals 1-2
Topic or Unit of Study:	Integrated Unit
Grade/Level:	9-12

Objective:

At the conclusion of this lesson students will be able to:

1. Add an existing Class (NamedUrl) to a project.

2. Define class and object in very broad terms.

3. Instantiate a NamedUrl object.

4. Write code to access the Url property of the object.

Summary:

Students add a Class to their Computer Integration project that had been written but "for some reason" not included in their project. They will use the class to store human-oriented names corresponding to computer-oriented URLs which contain information pertinent to their research. They create enough objects to contain their data and add them to a ComboBox in a subsequent lesson. For the time being, instructions are provided for testing their object without user input.

IMPLEMENTATION

Learning Context:

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. They are likely to be adding a ComboBox next.

Procedure:

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 how to add an existing item to a project, where to find the code, what a class and object are, how to create new objects, and how to access the stored URL. 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. The data used to initialize the objects will be different for every student as will what they do with it, although each will have to access it the same way.

Sample Student Products: The outward appearance of the program should not change in this lesson, although the project structure will.

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 as is the Visual Basic file which is edited in this activity. Students work only in Code View on this activity.

Attachments

1. **Class**
2. **NamedUrl**

Resources:

- Technology resources:
  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
      - **Competency :** 31.C EMPLOY MODULARITY IN WRITING PROGRAMS
        - **Indicator :** 31.2c Utilize parameters to pass data into program modules
      - **Competency :** 38.C EMPLOY OBJECT-ORIENTED PROGRAMMING TECHNIQUES
        - **Indicator :** 38.4c Write appropriate statements to invoke an object’s

accessor method(s)

- **Competency :** 31.C EMPLOY MODULARITY IN WRITING PROGRAMS
  - **Indicator :** 31.4c Write and use modules that return values
- **Competency :** 27.C DEMONSTRATE PROGRAM ANALYSIS AND DESIGN
  - **Indicator :** 27.9c Explain essential object analysis and design concepts
- **Competency :** 34.C USE SIMPLE DATA TYPES AND STRINGS
  - **Indicator :** 34.3c Declare and use constants in a program
- **Competency :** 38.C EMPLOY OBJECT-ORIENTED PROGRAMMING TECHNIQUES
  - **Indicator :** 38.1c Make a distinction between an object and a class
  - **Indicator :** 38.3c Instantiate objects from existing classes

Assessment/Rubrics:

By the end of the class period, students should have added the Class from file NamedUrl.vb. By the end of the unit (e.g., after the ComboBox activity), students should have written code to instantiate a number of NamedUrl objects and extracted the URL from them. Each activity contributes a few criteria to the larger rubric. This activity adds points for project structure, instantiation, and object use as described by the Class rubric.

Rubrics

1. **Class**