1 Understanding Data: Simple Classes

The theme for the first set of exercises will be the photo images a user can save in iPhoto, or other similar application. In this lab we define the classes that represent one photo in our collection of pictures, and at the end of the lab also add the option of saving video clips.

1.1 Data Definitions in HtDP

1. Log in and start *DrScheme*. Set the language to *Beginner HtDP*.

2. Design the data that represents a photo image. It should inlcude a name, the kind of graphics encoding (the file suffix, such as png, jpeg, tiff), the dimensions (the width and the height) and the size of the file in bytes.

   Of course, you remember that every data definition comes with examples of data.

3. Add examples of data that represent the following files:

   - Picture of a river (jpeg) that is 3456 pixels wide and 2304 pixels high, using up 3,614,571 bytes.
   - Picture of a mountain (jpeg) that is 2448 pixels wide and 3264 pixels high, using up 1,276,114 bytes.
   - Picture of a group of people (gif) that is 545 pixels wide and 641 pixels high, using up 13,760 bytes.
   - Picture of a plt icon (bmp) that is 16 pixels wide and 16 pixels high, using up 1334 bytes.

Before you proceed, please give to the TA your list of partner choices.

1.2 Data Definitions in ProfessorJ

Open a new Tab and set the language to *Beginner ProfessorJ*.

1. Design the Java class to represent an iPhoto picture. Name the class *Photo*. Create a class *Examples* at the end of your *Definitions* as follows:

```
class Examples {
  Examples() {}

  // examples of photo data
  Photo p1 = ...
  Photo p2 = ...
}
```

You will be defining all your sample data in the *Examples* class. Later, it will also be a place where you put all your tests.

Include in the *Examples* class definitions of all data that you have defined in the HtDP Beginner language.

2. Once you are done, run the program. You should see an instance of the *Examples* class in the *Interactions* window.

1.3 Data With Containment; Using the Wizard

Now that you have done the tedious typing, but understand what are the parts of a Java class definition, you can rewrite all your data definitions (except for the *Examples* class) using the *Wizard*. Observe first the similarities between the two data definitions. To define a Java class we only need to know the name of the class, its purpose, and for each field its *type* and its name. This information completely specifies what will be included in the class definition.

1. Start by looking at the classes that represent the date and the clock time. The *Date* class is defined in the text (page 13). The *ClockTime* class is defined on page 21.

   Open a new Tab using the *Beginner ProfessorJ* language. In the *Special* menu select *Insert Java Class*. Type in the class name *ClockTime*, add the purpose statement and the field information. Check the *add class diagram* checkbox. The code and the diagram is generated for you. (Do not check the *add toString method* checkbox!)

   Repeat these steps to define the *Date* class, but add a field that represents the day of the week (e.g. "Friday").

   Now define the *Examples* class and make examples of *ClockTime* and *Date* data.

2. The photo file information also includes the time when the photo file was created. Design a new *Photo* class that includes this information.

Again, use the *Insert Java Class* wizard to generate the class diagram and the Java code.

3. Adjust the class diagram by including the diagrams for the *Date* class and for the *ClockTime* class with the appropriate *containment* arrows.

4. Make sure to include examples of the instances of the new *Photo* class in your *Examples* class.

5. Write down the data definitions for these classes of data in the *Beginner HtDP* language as well.

Stop and see who is the partner you have been assigned. Meet your partner - exchange the names, phone numbers, and schedule the first meeting. Work on the rest of the lab together.

1.4 Unions of Data

You now have a new camera that allows you to take not only still pictures, but also short videos. You want to keep the two kinds of files together. Here is a data definition for the data you may want - written in the *Beginner HtDP* language:

```
;; Data to represent a camera shot:
;; either a still photo or a video clip

;; A Shot is one of
;; – Photo
;; – Video

;; we omit the definition of the Photo
;; as you already have done that

;; A Video is (make-video String String Number Number Boolean)
(define-struct video (name kind size duration sound?))
;; Interpretation:
;; kind of video may be either QuickTime or RealPlayer
;; size is measured in bytes
;; duration is measured in seconds
;; sound? indicated whether or not this video has sound
```

1. Define the Java classes that correspond to the given data definition by sketching the class diagram on a paper. Also, make examples of data - by hand.

2. Add the field that represents the time when the video was created to the class Video. Now use the *Insert Java Union* wizard to convert the data definitions to class diagrams and Java code.

3. **Do we have to remind you to make examples of data???**

Save all your work — the next lab will build on the work you have done here!

If you have some time left, work on the *Etudes* part of the homework.