```
  Course: R Programming
  Lesson: Base Graphics


- Class: text
  Output: One of the greatest strengths of R, relative to other programming languages,
  is the ease with which we can create publication-quality graphics. In this lesson,
  you'll learn about base graphics in R.

- Class: text
  Output: We do not cover the more advanced portions of graphics in R in this lesson.
  These include lattice, ggplot2 and ggvis.

- Class: text
  Output: There is a school of thought that this approach is backwards, that we should
  teach ggplot2 first. See http://varianceexplained.org/r/teach_ggplot2_to_beginners/
  for an outline of this view.

- Class: cmd_question
  Output: Load the included data frame cars with data(cars).
  CorrectAnswer: data(cars)
  AnswerTests: omnitest(correctExpr='data(cars)')
  Hint: Type data(cars) to load the data.

- Class: text
  Output: To fix ideas, we will work with simple data frames. Our main goal is to
  introduce various plotting functions and their arguments. All the output would look
  more interesting with larger, more complex data sets.

- Class: cmd_question
  Output: Pull up the help page for cars.
  CorrectAnswer: ?cars
  AnswerTests: any_of_exprs('?cars', 'help(cars)', 'help("plot")')
  Hint: Type ?cars or help(cars) to view a help page with details on the car data frame.

- Class: text
  Output: "As you can see in the help page, the cars data set has only two variables:
  speed and stopping distance. Note that the data is from the 1920s."

- Class: cmd_question
  Output: Run head() on the cars data.
  CorrectAnswer: head(cars)
  AnswerTests: omnitest(correctExpr='head(cars)')
  Hint: Type head(cars) to see the top of the cars data frame.

- Class: text
  Output: Before plotting, it is always a good idea to get a sense of the data. Key R
  commands for doing so include, dim(), names(), head(), tail() and summary().

- Class: cmd_question
  Output: Run the plot() command on the cars data frame.
  CorrectAnswer: plot(cars)
  AnswerTests: omnitest(correctExpr='plot(cars)')
  Hint: Type plot(cars) to create a plot of the cars data frame.

- Class: text
  Output: As always, R tries very hard to give you something sensible given the
  information that you have provided to it. First, R notes that the data frame you have
  given it has just two columns, so it assumes that you want to plot one column versus
  the other.

- Class: text
  Output: Second, since we do not provide labels for either axis, R uses the names of
  the columns. Third, it creates axis tick marks at nice round numbers and labels them
  accordingly. Fourth, it uses the other defaults supplied in plot().

- Class: text
  Output: We will now spend some time exploring plot, but many of the topics covered
  here will apply to most other R graphics functions. Note that 'plot' is short for
```

```
    scatterplot.

- Class: cmd_question
  Output: Look up the help page for plot().
  CorrectAnswer: ?plot
  AnswerTests: any_of_exprs('?plot', 'help(plot)')
  Hint: Type ?plot or help(plot) to view a help page for plot().

- Class: text
  Output: The help page for plot() highlights the different arguments that the function can take. The two most important are x and y, the variables that will be plotted. For the next set of questions, include the argument names in your answers. That is, do not type plot(cars$speed, cars$dist), although that will work. Instead, use plot(x = cars$speed, y = cars$dist).

- Class: cmd_question
  Output: Use plot() command to show speed on the x-axis and dist on the y-axis from the cars data frame. Use the form of the plot command in which vectors are explicitly passed in as arguments for x and y.
  CorrectAnswer: plot(x = cars$speed, y = cars$dist)
  AnswerTests: omnitest(correctExpr='plot(x = cars$speed, y = cars$dist)')
  Hint: Type plot(x = cars$speed, y = cars$dist) to create the plot.

- Class: text
  Output: Note that this produces a slightly different answer than plot(cars). In this case, R is not sure what you want to use as the labels on the axes, so it just uses the arguments which you pass in, data frame name and dollar signs included.

- Class: text
  Output: Note that there are other ways to call the plot command, i.e., using the "formula" interface. For example, we get a similar plot to the above with plot(dist ~ speed, cars). However, we will wait till later in the lesson before using the formula interface.

- Class: cmd_question
  Output: Use plot() command to show dist on the x-axis and speed on the y-axis from the cars data frame. This is the opposite of what we did above.
  CorrectAnswer: plot(x = cars$dist, y = cars$speed)
  AnswerTests: omnitest(correctExpr='plot(x = cars$dist, y = cars$speed)')
  Hint: Type plot(x = cars$dist, y = cars$speed) to create the plot.

- Class: text
  Output: It probably makes more sense for speed to go on the x-axis since stopping distance is a function of speed more than the other way around. So, for the rest of the questions in this portion of the lesson, always assign the arguments accordingly.

- Class: text
  Output: In fact, you can assume that the answers to the next few questions are all of the form plot(x = cars$speed, y = cars$dist, ...) but with various arguments used in place of the ...

- Class: cmd_question
  Output: Recreate the plot with the label of the x-axis set to "Speed".
  CorrectAnswer: plot(x = cars$speed, y = cars$dist, xlab = "Speed")
  AnswerTests: omnitest(correctExpr='plot(x = cars$speed, y = cars$dist, xlab = "Speed")')
  Hint: Type plot(x = cars$speed, y = cars$dist, xlab = "Speed") to create the plot.

- Class: cmd_question
  Output: Recreate the plot with the label of the y-axis set to "Stopping Distance".
  CorrectAnswer: plot(x = cars$speed, y = cars$dist, ylab = "Stopping Distance")
  AnswerTests: omnitest(correctExpr='plot(x = cars$speed, y = cars$dist, ylab = "Stopping Distance")')
  Hint: Type plot(x = cars$speed, y = cars$dist, ylab = "Stopping Distance") to create the plot.

- Class: cmd_question
  Output: Recreate the plot with "Speed" and "Stopping Distance" as axis labels.
  CorrectAnswer: plot(x = cars$speed, y = cars$dist, xlab = "Speed", ylab = "Stopping
```

```
    Distance")
    AnswerTests: omnitest(correctExpr='plot(x = cars$speed, y = cars$dist, xlab =
    "Speed", ylab = "Stopping Distance")')
    Hint: Type plot(x = cars$speed, y = cars$dist, xlab = "Speed", ylab = "Stopping
    Distance") to create the plot.

- Class: text
  Output: The reason that plots(cars) worked at the beginning of the lesson was that R
  was smart enough to know that the first element (i.e., the first column) in cars
  should be assigned to the x argument and the second element to the y argument. To
  save on typing, the next set of answers will all be of the form, plot(cars, ...) with
  various arguments added.

- Class: text
  Output: For each question, we will only want one additional argument at a time. Of
  course, you can pass in more than one argument when doing a real project.

- Class: cmd_question
  Output: Plot cars with a main title of "My Plot". Note that the argument for the main
  title is "main" not "title".
  CorrectAnswer: plot(cars, main = "My Plot")
  AnswerTests: omnitest(correctExpr='plot(cars, main = "My Plot")')
  Hint: Type plot(cars, main = "My Plot") to create the plot.

- Class: cmd_question
  Output: Plot cars with a sub title of "My Plot Subtitle".
  CorrectAnswer: plot(cars, sub = "My Plot Subtitle")
  AnswerTests: omnitest(correctExpr='plot(cars, sub = "My Plot Subtitle")')
  Hint: Type plot(cars, sub = "My Plot Subtitle") to create the plot.

- Class: text
  Output: The plot help page (?plot) only covers a small number of the many arguments
  that can be passed in to plot() and to other graphical functions. To begin to explore
  the many other options, look at ?par. Let's look at some of the more commonly used
  ones. Continue using plot(cars, ...) as the base answer to these questions.

- Class: cmd_question
  Output: Plot cars so that the plotted points are colored red. (Use col = 2 to achieve
  this effect.)
  CorrectAnswer: plot(cars, col = 2)
  AnswerTests: omnitest(correctExpr='plot(cars, col = 2)')
  Hint: Type plot(cars, col = 2) to create the plot.

- Class: cmd_question
  Output: Plot cars while limiting the x-axis to 10 through 15.  (Use xlim = c(10, 15)
  to achieve this effect.)
  CorrectAnswer: plot(cars, xlim = c(10, 15))
  AnswerTests: omnitest(correctExpr='plot(cars, xlim = c(10, 15))')
  Hint: Type plot(cars, xlim = c(10, 15)) to create the plot.

- Class: text
  Output: You can also change the shape of the symbols in the plot. The help page for
  points (?points) provides the details.

- Class: cmd_question
  Output: Plot cars using triangles.  (Use pch = 2 to achieve this effect.)
  CorrectAnswer: plot(cars, pch = 2)
  AnswerTests: omnitest(correctExpr='plot(cars, pch = 2)')
  Hint: Type plot(cars, pch = 2) to create the plot.

- Class: text
  Output: Arguments like "col" and "pch" may not seem very intuitive. And that is
  because they aren't! So, many/most people use more modern packages, like ggplot2, for
  creating their graphics in R.

- Class: text
  Output: It is, however, useful to have an introduction to base graphics because many
  of the idioms in lattice and ggplot2 are modeled on them.

```

```
- Class: text
  Output: Let's now look at some other functions in base graphics that may be useful,
  starting with boxplots.

- Class: cmd_question
  Output: Load the mtcars data frame.
  CorrectAnswer: data(mtcars)
  AnswerTests: omnitest(correctExpr='data(mtcars)')
  Hint: Type data(mtcars) to load the data.

- Class: text
  Output: Anytime that you load up a new data frame, you should explore it before using
  it. In the middle of a swirl lesson, just type play(). This temporarily suspends the
  lesson (without losing the work you have already done) and allows you to issue
  commands like dim(mtcars) and head(mtcars). Once you are done examining the data,
  just type nxt() and the lesson will pick up where it left off.

- Class: cmd_question
  Output: Look up the help page for boxplot().
  CorrectAnswer: ?boxplot
  AnswerTests: any_of_exprs('?boxplot', 'help(boxplot)')
  Hint: Type ?boxplot or help(boxplot) to view a help page with details about boxplot.

- Class: text
  Output: Instead of adding data columns directly as input arguments, as we did with
  plot(), it is often handy to pass in the entire data frame. This is what the "data"
  argument in boxplot() allows.

- Class: text
  Output: boxplot(), like many R functions, also takes a "formula" argument, generally
  an expression with a tilde ("~") which indicates the relationship between the input
  variables. This allows you to enter something like mpg ~ cyl to plot the relationship
  between cyl (number of cylinders) on the x-axis and mpg (miles per gallon) on the
  y-axis.

- Class: cmd_question
  Output: Use boxplot() with formula = mpg ~ cyl and data = mtcars to create a box plot.
  CorrectAnswer: boxplot(formula = mpg ~ cyl, data = mtcars)
  AnswerTests: omnitest(correctExpr='boxplot(formula = mpg ~ cyl, data = mtcars)')
  Hint: Type boxplot(formula = mpg ~ cyl, data = mtcars) to create the plot.

- Class: text
  Output: The plot shows that mpg is much lower for cars with more cylinders. Note that
  we can use the same set of arguments that we explored with plot() above to add axis
  labels, titles and so on.

- Class: text
  Output: When looking at a single variable, histograms are a useful tool. hist() is
  the associated R function. Like plot(), hist() is best used by just passing in a
  single vector.

- Class: cmd_question
  Output: Use hist() with the vector mtcars$mpg to create a histogram.
  CorrectAnswer: hist(mtcars$mpg)
  AnswerTests: any_of_exprs('hist(mtcars$mpg)', 'hist(x = mtcars$mpg)')
  Hint: Type hist(mtcars$mpg) to create the plot.

# Not sure what a good lesson length is for this.
# Might add some information on saving plots.
# Other functions that I use include identify().

- Class: text
  Output: In this lesson, you learned how to work with base graphics in R. The best
  place to go from here is to study the ggplot2 package. If you want to explore other
  elements of base graphics, then this web page
  (http://www.ling.upenn.edu/~joseff/rstudy/week4.html) provides a useful overview.

- Class: mult_question
  Output: "Would you like to receive credit for completing this course on
```

```
206       Coursera.org?"
207     CorrectAnswer: NULL
208     AnswerChoices: Yes;No
209     AnswerTests: coursera_on_demand()
210     Hint: ""
211
```