2.9 The Structure of Graphics and Sound

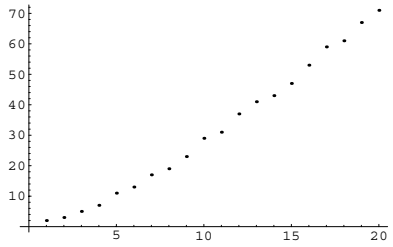
2.9.1 The Structure of Graphics

Section 1.9 discussed how to use functions like `Plot` and `ListPlot` to plot graphs of functions and data. In this section, we discuss how *Mathematica* represents such graphics, and how you can program *Mathematica* to create more complicated images.

The basic idea is that *Mathematica* represents all graphics in terms of a collection of *graphics primitives*. The primitives are objects like `Point`, `Line` and `Polygon`, that represent elements of a graphical image, as well as directives such as `RGBColor`, `Thickness` and `SurfaceColor`.

This generates a plot of a list of points.

```
In[1]:= ListPlot[ Table[Prime[n], {n, 20}] ]
```



`InputForm` shows how *Mathematica* represents the graphics. Each point is represented as a `Point` graphics primitive. All the various graphics options used in this case are also given.

```
In[2]:= InputForm[%]

Out[2]//InputForm=
  Graphics[{Point[{1, 2}], Point[{2, 3}], Point[{3, 5}],
    Point[{4, 7}], Point[{5, 11}], Point[{6, 13}],
    Point[{7, 17}], Point[{8, 19}], Point[{9, 23}],
    Point[{10, 29}], Point[{11, 31}], Point[{12, 37}],
    Point[{13, 41}], Point[{14, 43}], Point[{15, 47}],
    Point[{16, 53}], Point[{17, 59}], Point[{18, 61}],
    Point[{19, 67}], Point[{20, 71}]},
   {PlotRange -> Automatic, AspectRatio -> GoldenRatio^(-1),
    DisplayFunction :> $DisplayFunction,
    ColorOutput -> Automatic, Axes -> Automatic,
    AxesOrigin -> Automatic, PlotLabel -> None,
    AxesLabel -> None, Ticks -> Automatic, GridLines -> None,
    Prolog -> {}, Epilog -> {}, AxesStyle -> Automatic,
    Background -> Automatic, DefaultColor -> Automatic,
    DefaultFont :> $DefaultFont, RotateLabel -> True,
    Frame -> False, FrameStyle -> Automatic,
    FrameTicks -> Automatic, FrameLabel -> None,
    PlotRegion -> Automatic}]
```

Each complete piece of graphics in *Mathematica* is represented as a *graphics object*. There are several different kinds of graphics object, corresponding to different types of graphics. Each kind of graphics object has a definite head which identifies its type.

Web sample page from The *Mathematica* Book, Second Edition, by Stephen Wolfram, published by Addison-Wesley Publishing Company (hardcover ISBN 0-201-51502-4; softcover ISBN 0-201-51507-5). To order *Mathematica* or this book contact Wolfram Research: info@wolfram.com; http://www.wolfram.com/; 1-800-441-6284.

`Graphics[`*list*`]`	general two-dimensional graphics
`DensityGraphics[`*list*`]`	density plot
`ContourGraphics[`*list*`]`	contour plot
`SurfaceGraphics[`*list*`]`	three-dimensional surface
`Graphics3D[`*list*`]`	general three-dimensional graphics
`GraphicsArray[`*list*`]`	array of other graphics objects

Graphics objects in *Mathematica*.

The functions like `Plot` and `ListPlot` discussed in Section 1.9 all work by building up *Mathematica* graphics objects, and then displaying them.

`Graphics`	`Plot, ListPlot, ParametricPlot`
`DensityGraphics`	`DensityPlot, ListDensityPlot`
`ContourGraphics`	`ContourPlot, ListContourPlot`
`SurfaceGraphics`	`Plot3D, ListPlot3D`
`Graphics3D`	`ParametricPlot3D`

Generating graphics objects by plotting functions and data.

You can create other kinds of graphical images in *Mathematica* by building up your own graphics objects. Since graphics objects in *Mathematica* are just symbolic expressions, you can use all the standard *Mathematica* functions to manipulate them.

Once you have created a graphics object, you must then display it. The function `Show` allows you to display any *Mathematica* graphics object.

`Show[`g`]`	display a graphics object
`Show[`g_1`, `g_2`, ... ]`	display several graphics objects combined
`Show[GraphicsArray[{{`g_{11}`, `g_{12}`, ... }, ... }]]`	display an array of graphics objects

Displaying graphics objects.

This uses `Table` to generate a polygon graphics primitive.

```
In[3]:= poly = Polygon[
          Table[N[{Cos[n Pi/5], Sin[n Pi/5]}], {n, 0, 5}] ]
Out[3]= Polygon[{{1., 0}, {0.809017, 0.587785},
   {0.309017, 0.951057}, {-0.309017, 0.951057},
   {-0.809017, 0.587785}, {-1., 0}}]
```

This creates a two-dimensional graphics object that contains the polygon graphics primitive. In standard output format, the graphics object is given simply as `-Graphics-`.

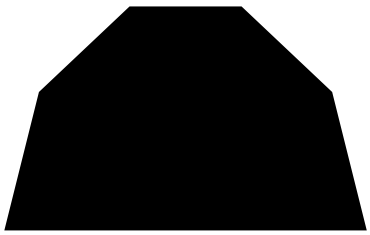
```
In[4]:= Graphics[ poly ]
Out[4]= -Graphics-
```

`InputForm` shows the complete graphics object.

```
In[5]:= InputForm[%]
Out[5]//InputForm=
  Graphics[Polygon[{{1., 0},
     {0.8090169943749474, 0.5877852522924732},
     {0.3090169943749474, 0.9510565162951536},
     {-0.3090169943749474, 0.9510565162951536},
     {-0.8090169943749474, 0.5877852522924733}, {-1., 0}}]]
```

This displays the graphics object you have created.

```
In[6]:= Show[%]
```



Graphics directives	Examples: `RGBColor`, `Thickness`, `SurfaceColor`
Graphics options	Examples: `PlotRange`, `Ticks`, `AspectRatio`, `ViewPoint`

Local and global ways to modify graphics.

Given a particular list of graphics primitives, *Mathematica* provides two basic mechanisms for modifying the final form of graphics you get. First, you can insert into the list of graphics primitives certain *graphics directives*, such as `RGBColor`, which modify the subsequent graphical elements in the list. In this way, you can specify how a particular set of graphical elements should be rendered.

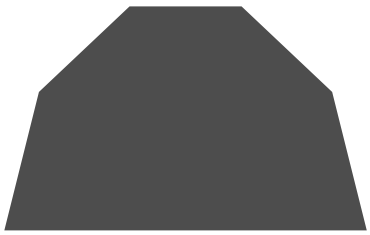
This takes the list of graphics primitives created above, and adds the graphics directive `GrayLevel[0.3]`.

```
In[7]:= Graphics[ {GrayLevel[0.3], poly} ]
Out[7]= -Graphics-
```

Web sample page from The *Mathematica* Book, Second Edition, by Stephen Wolfram, published by Addison-Wesley Publishing Company (hardcover ISBN 0-201-51502-4; softcover ISBN 0-201-51507-5). To order *Mathematica* or this book contact Wolfram Research: info@wolfram.com; http://www.wolfram.com/; 1-800-441-6284.

Now the polygon is rendered in gray.

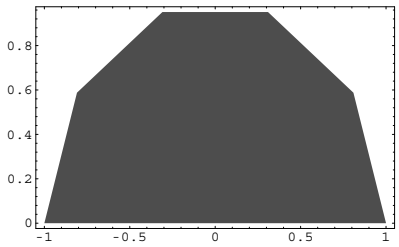
```
In[8]:= Show[%]
```



By inserting graphics directives, you can specify how particular graphical elements should be rendered. Often, however, you want to make global modifications to the way a whole graphics object is rendered. You can do this using *graphics options*.

By adding the graphics option `Frame` you can modify the overall appearance of the graphics.

```
In[9]:= Show[%, Frame -> True]
```



`Show` returns a graphics object with the options in it.

```
In[10]:= InputForm[%]

Out[10]//InputForm=
  Graphics[{GrayLevel[0.3],
    Polygon[{{1., 0}, {0.809016994374947, 0.5877852522924732},
      {0.3090169943749474, 0.951056516295153},
      {-0.3090169943749474, 0.951056516295154},
      {-0.809016994374947, 0.5877852522924733}, {-1., 0}}]},
   Frame -> True]
```

You can specify graphics options in `Show`. As a result, it is straightforward to take a single graphics object, and show it with many different choices of graphics options.

Notice however that `Show` always returns the graphics objects it has displayed. If you specify graphics options in `Show`, then these options are automatically inserted into the graphics objects that `Show` returns. As a result, if you call `Show` again on the same objects, the same graphics options will be used, unless you explicitly specify other ones. Note that in all cases new options you specify will overwrite ones already there.

Web sample page from The *Mathematica* Book, Second Edition, by Stephen Wolfram, published by Addison-Wesley Publishing Company (hardcover ISBN 0-201-51502-4; softcover ISBN 0-201-51507-5). To order *Mathematica* or this book contact Wolfram Research: info@wolfram.com; http://www.wolfram.com/; 1-800-441-6284.

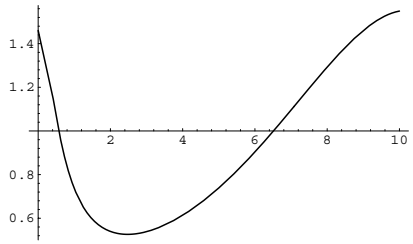
`Options[`*g*`]`	give a list of all graphics options for a graphics object
`Options[`*g*`, `*opt*`]`	give the setting for a particular option
`FullOptions[`*g*`, `*opt*`]`	give the actual value used for a particular option, even if the setting is `Automatic`

Finding the options for a graphics object.

Some graphics options work by requiring you to specify a particular value for a parameter related to a piece of graphics. Other options allow you to give the setting `Automatic`, which makes *Mathematica* use internal algorithms to choose appropriate values for parameters. In such cases, you can find out the values that *Mathematica* actually used by applying the function `FullOptions`.

Here is a plot.

```
In[11]:= zplot = Plot[Abs[Zeta[1/2 + I x]], {x, 0, 10}]
```



The option `PlotRange` is set to its default value of `Automatic`, specifying that *Mathematica* should use internal algorithms to determine the actual plot range.

```
In[12]:= Options[zplot, PlotRange]
Out[12]= {PlotRange -> Automatic}
```

`FullOptions` gives the actual plot range determined by *Mathematica* in this case.

```
In[13]:= FullOptions[zplot, PlotRange]
Out[13]= {{-0.25, 10.25}, {0.50068, 1.57477}}
```

`FullGraphics[`*g*`]`	translate objects specified by graphics options into lists of explicit graphics primitives

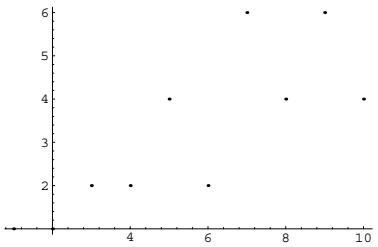
Finding the complete form of a piece of graphics.

When you use a graphics option such as `Axes`, *Mathematica* effectively has to construct a list of graphics elements to represent the objects such as axes that you have requested. Usually *Mathematica* does not explicitly return the list it constructs in this way. Sometimes, however, you may find it useful to get this list. The function `FullGraphics` gives the complete list of graphics primitives needed to generate a particular plot, without any options being used.

Web sample page from The *Mathematica* Book, Second Edition, by Stephen Wolfram, published by Addison-Wesley Publishing Company (hardcover ISBN 0-201-51502-4; softcover ISBN 0-201-51507-5). To order *Mathematica* or this book contact Wolfram Research: info@wolfram.com; http://www.wolfram.com/; 1-800-441-6284.

This plots a list of values.

```
In[14]:= ListPlot[ Table[EulerPhi[n], {n, 10}] ]
```



`FullGraphics` yields a graphics object that includes graphics primitives representing axes and so on.

```
In[15]:= Short[ InputForm[ FullGraphics[%] ], 6]
Out[15]//Short=
 Graphics[{{Point[{1, 1}], Point[{2, 1}], Point[{3, 2}],
    Point[{4, 2}], Point[{5, 4}], Point[{6, 2}], Point[{7, 6}],
    Point[{8, 4}], Point[{9, 6}], Point[{10, 4}]},
   {{{{GrayLevel[0.], Thickness[0.002],
       Line[{{0.775, 1.}, {10.225, 1.}}]}, None}}, <<2>>,
    {{<<2>>}, <<24>>}}}]
```

With their default option settings, functions like `Plot` and `Show` actually cause *Mathematica* to generate graphical output. In general, the actual generation of graphical output is controlled by the graphics option `DisplayFunction`. The default setting for this option is the value of the global variable `$DisplayFunction`.

In most cases, `$DisplayFunction` and the `DisplayFunction` option are set to use the lower-level rendering function `Display` to produce output, perhaps after some preprocessing. Sometimes, however, you may want to get a function like `Plot` to produce a graphics object, but you may not immediately want that graphics object actually rendered as output. You can tell *Mathematica* to generate the object, but not render it, by setting the option `DisplayFunction -> Identity`. Section 2.9.14 will explain exactly how this works.

`Plot[`*f*`, ... , DisplayFunction -> Identity]`, etc.	generate a graphics object for a plot, but do not actually display it
`Show[`*g*`, DisplayFunction -> $DisplayFunction]`	show a graphics object using the default display function

Generating and displaying graphics objects.

Web sample page from The *Mathematica* Book, Second Edition, by Stephen Wolfram, published by Addison-Wesley Publishing Company (hardcover ISBN 0-201-51502-4; softcover ISBN 0-201-51507-5). To order *Mathematica* or this book contact Wolfram Research: info@wolfram.com; http://www.wolfram.com/; 1-800-441-6284.

This generates a graphics object, but does not actually display it.

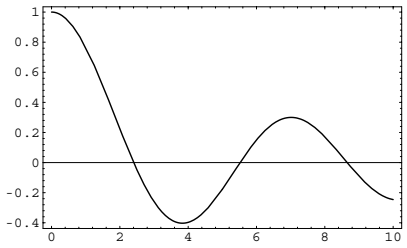
```
In[16]:= Plot[BesselJ[0, x], {x, 0, 10},
                              DisplayFunction -> Identity]
Out[16]= -Graphics-
```

This modifies the graphics object, but still does not actually display it.

```
In[17]:= Show[%, Frame -> True]
Out[17]= -Graphics-
```

To display the graphic, you explicitly have to tell *Mathematica* to use the default display function.

```
In[18]:= Show[%, DisplayFunction -> $DisplayFunction]
```



Web sample page from The *Mathematica* Book, Second Edition, by Stephen Wolfram, published by Addison-Wesley Publishing Company (hardcover ISBN 0-201-51502-4; softcover ISBN 0-201-51507-5). To order *Mathematica* or this book contact Wolfram Research: info@wolfram.com; http://www.wolfram.com/; 1-800-441-6284.