



Visualisation with `ggplot2`

R Foundations for Life Scientists

Roy Francis

Why `ggplot2`?



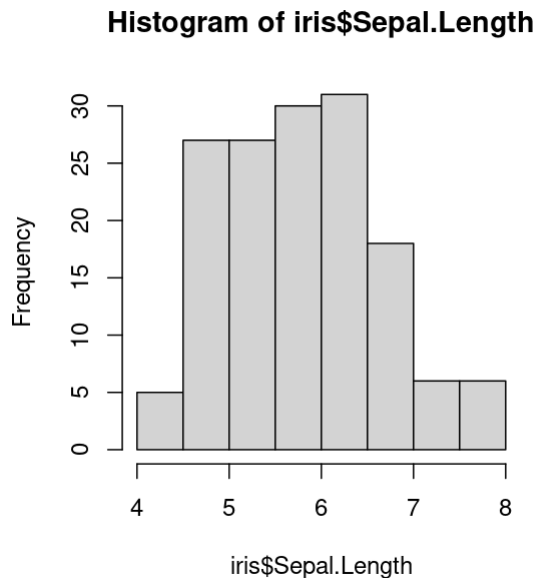
- Consistent code
- Flexible
- Automatic legends, colors etc
- Save plot objects
- Themes for reusing styles
- Numerous add-ons/extensions
- Nearly complete graphing solution

Not suitable for:

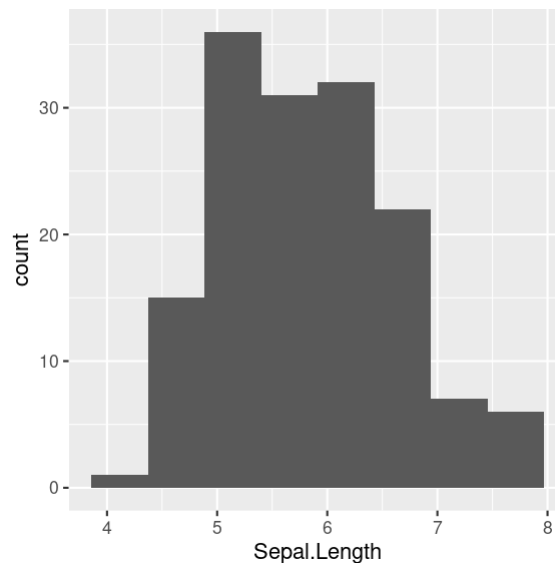
- 3D graphics

ggplot2 vs Base Graphics

```
hist(iris$Sepal.Length)
```

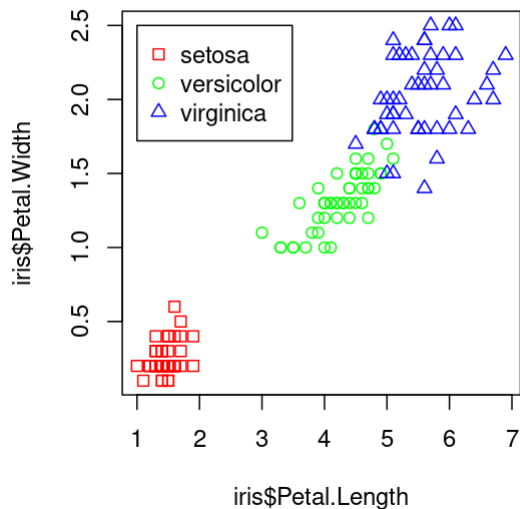


```
library(ggplot2)  
ggplot(iris, aes(x=Sepal.Length)) +  
  geom_histogram(bins=8)
```

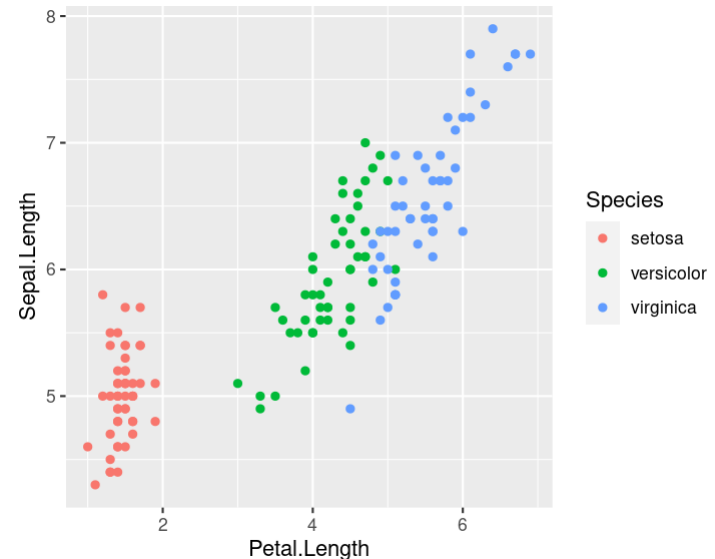


ggplot2 vs Base Graphics

```
plot(iris$Petal.Length,iris$Petal.Width,  
     col=c("red","green","blue")[iris$Species],  
     pch=c(0,1,2)[iris$Species])  
legend(x=1,y=2.5,  
       legend=c("setosa","versicolor","virginica"),  
       pch=c(0,1,2),col=c("red","green","blue"))
```



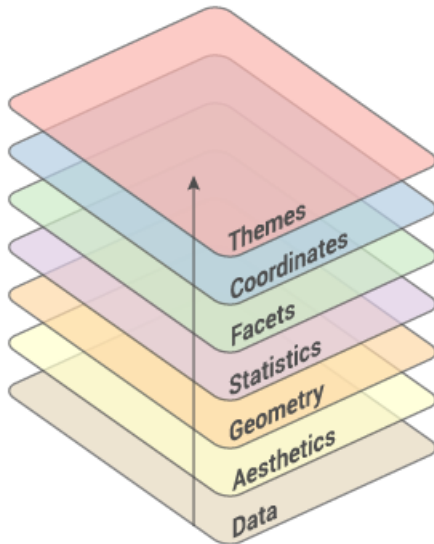
```
ggplot(iris,aes(Petal.Length,Sepal.Length,color=Species))  
geom_point()
```



Grammar Of Graphics



- **Data:** Input data
- **Geom:** A geometry representing data. Points, Lines etc
- **Aesthetic:** Visual characteristics of the geometry. Size, Color, Shape etc
- **Scale:** How visual characteristics are converted to display values
- **Statistics:** Statistical transformations. Counts, Means etc
- **Coordinates:** Numeric system to determine position of geometry. Cartesian, Polar etc
- **Facets:** Split data into subsets



Building A Graph: Syntax

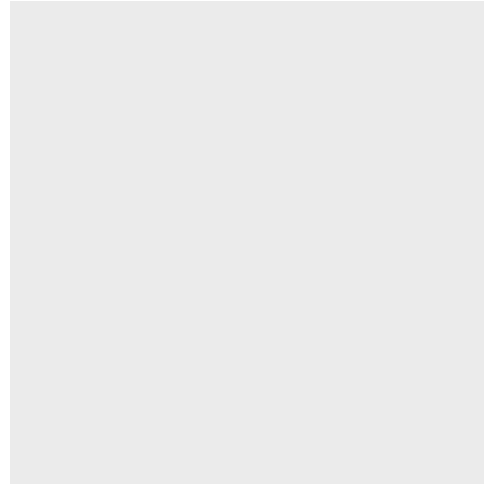
```
ggplot (data = <DATA> ) +  
  <GEOM_FUNCTION> (mapping = aes( <MAPPINGS> ),  
  stat = <STAT> , position = <POSITION> ) +  
  <COORDINATE_FUNCTION> +  
  <FACET_FUNCTION> +  
  <SCALE_FUNCTION> +  
  <THEME_FUNCTION>
```

required

Not required, sensible defaults supplied

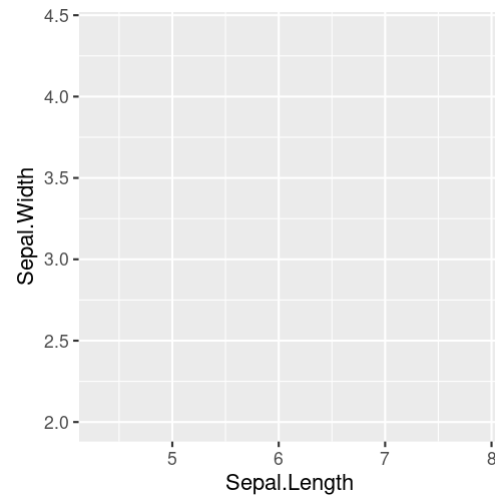
Building A Graph

```
ggplot(iris)
```



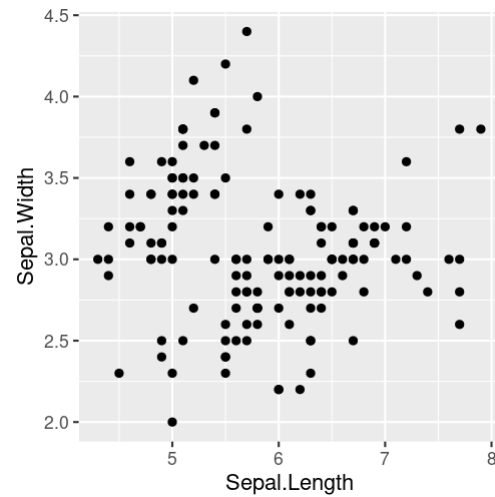
Building A Graph

```
ggplot(iris,aes(x=Sepal.Length,  
                y=Sepal.Width))
```



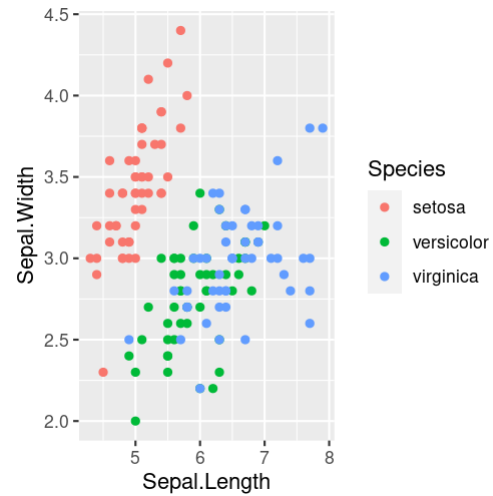
Building A Graph

```
ggplot(iris,aes(x=Sepal.Length,  
                y=Sepal.Width))+  
  geom_point()
```



Building A Graph

```
ggplot(iris,aes(x=Sepal.Length,  
                y=Sepal.Width,  
                colour=Species))+  
  geom_point()
```



Data • iris

- Input data is always an R `data.frame` object

| Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species |
|--------------|-------------|--------------|-------------|---------|
| 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| 4.7 | 3.2 | 1.3 | 0.2 | setosa |

```
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
## $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
## $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
## $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
## $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
## $ Species      : Factor w/ 3 levels "setosa","versicolor",...: 1 1 1 1 1 1 1 1 1 1 ...
```

Data • diamonds



| carat | cut | color | clarity | depth | table | price | x | y | z |
|-------|---------|-------|---------|-------|-------|-------|------|------|------|
| 0.23 | Ideal | E | SI2 | 61.5 | 55 | 326 | 3.95 | 3.98 | 2.43 |
| 0.21 | Premium | E | SI1 | 59.8 | 61 | 326 | 3.89 | 3.84 | 2.31 |
| 0.23 | Good | E | VS1 | 56.9 | 65 | 327 | 4.05 | 4.07 | 2.31 |

str(diamonds)

```
## tibble [53,940 × 10] (S3: tbl_df/tbl/data.frame)
## $ carat : num [1:53940] 0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
## $ cut   : Ord.factor w/ 5 levels "Fair"<"Good"<...: 5 4 2 4 2 3 3 3 1 3 ...
## $ color : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<...: 2 2 2 6 7 7 6 5 2 5 ...
## $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<...: 2 3 5 4 2 6 7 3 4 5 ...
## $ depth : num [1:53940] 61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
## $ table : num [1:53940] 55 61 65 58 58 57 57 55 61 61 ...
## $ price : int [1:53940] 326 326 327 334 335 336 336 337 337 338 ...
## $ x      : num [1:53940] 3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
## $ y      : num [1:53940] 3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
## $ z      : num [1:53940] 2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
```

Data • Format

- Transforming data into long or wide formats

```
iris %>% head(n=4)
```

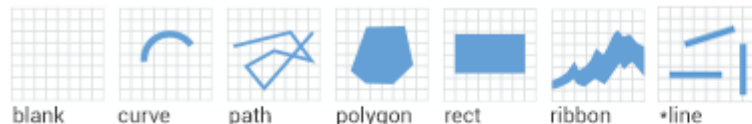
| | Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species |
|---|--------------|-------------|--------------|-------------|---------|
| 1 | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 2 | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| 3 | 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 4 | 4.6 | 3.1 | 1.5 | 0.2 | setosa |

```
iris %>% tidyr::pivot_longer(!Species,names_to="variable",values_to="value") %>%  
  as.data.frame() %>% head(n=5)
```

| | Species | variable | value |
|---|---------|--------------|-------|
| 1 | setosa | Sepal.Length | 5.1 |
| 2 | setosa | Sepal.Width | 3.5 |
| 3 | setosa | Petal.Length | 1.4 |
| 4 | setosa | Petal.Width | 0.2 |
| 5 | setosa | Sepal.Length | 4.9 |

Geoms

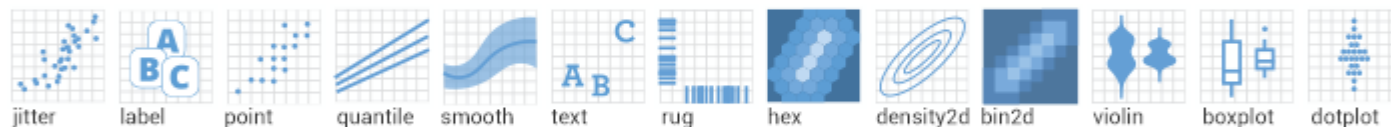
Basic



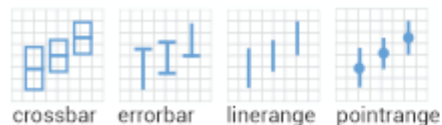
One variable



Two variables



Error



Three variables



Map

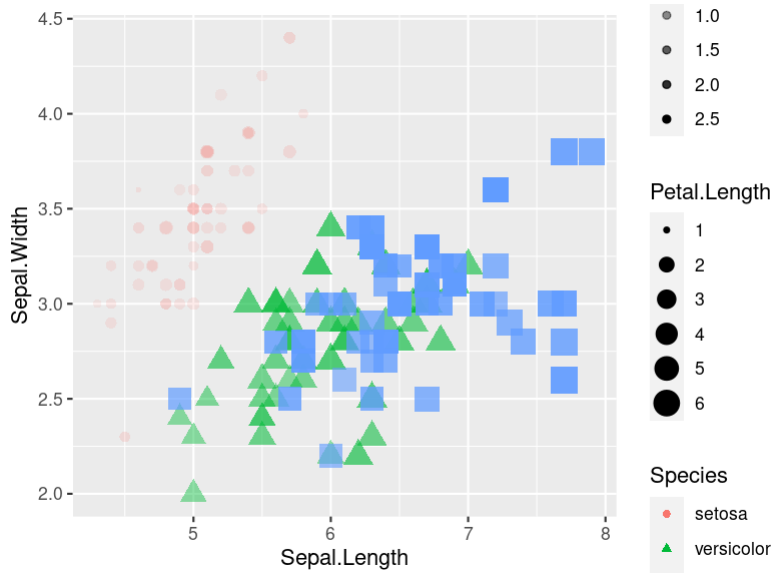


```
p <- ggplot(iris)
# scatterplot
p+geom_point(aes(x=Sepal.Length,y=Sepal.Width))
# barplot
p+geom_bar(aes(x=Sepal.Length))
# boxplot
p+geom_boxplot(aes(x=Species,y=Sepal.Width))
# search
help.search("^geom_",package="ggplot2")
```

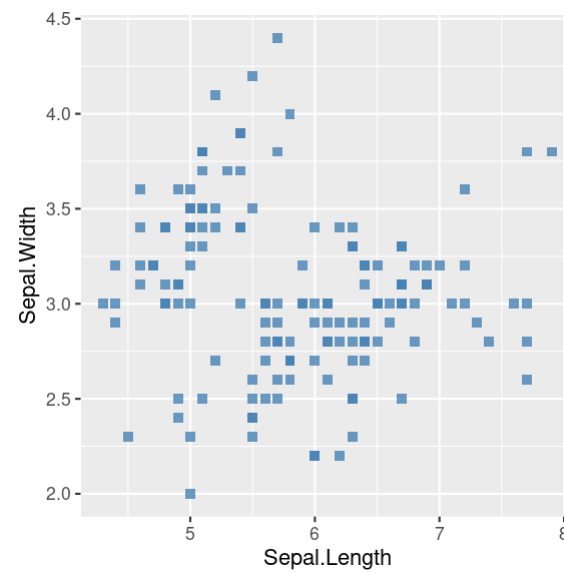
Aesthetics

- Aesthetic mapping vs aesthetic parameter

```
ggplot(iris)+  
  geom_point(aes(x=Sepal.Length,  
                 y=Sepal.Width,  
                 size=Petal.Length,  
                 alpha=Petal.Width,  
                 shape=Species,  
                 color=Species))
```



```
ggplot(iris)+  
  geom_point(aes(x=Sepal.Length,  
                 y=Sepal.Width),  
            size=2,  
            alpha=0.8,  
            shape=15,  
            color="steelblue")
```

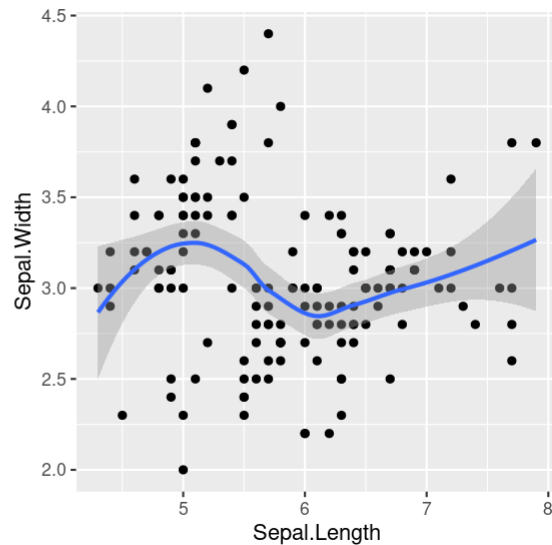
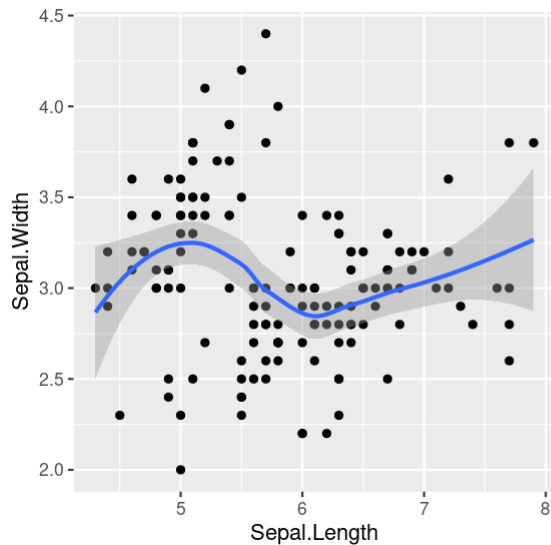


Aesthetics

```
x1 <- ggplot(iris) +  
  geom_point(aes(x=Sepal.Length,y=Sepal.Width))+  
  stat_smooth(aes(x=Sepal.Length,y=Sepal.Width))
```

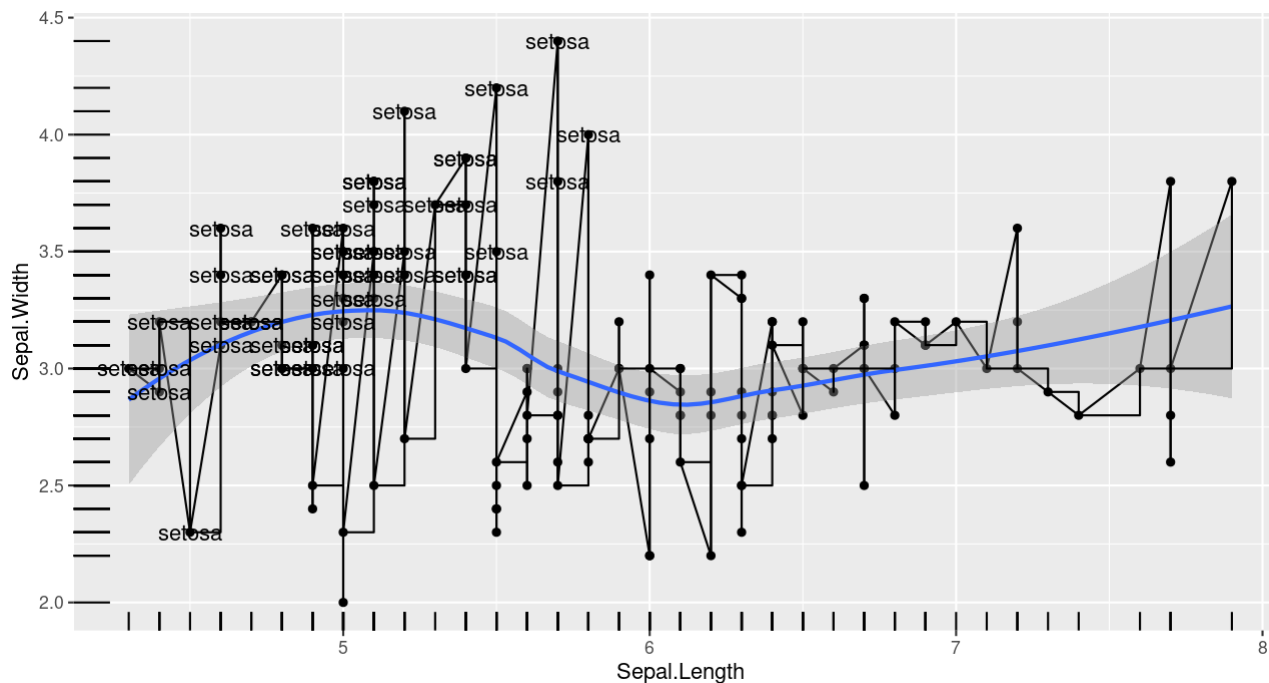
```
x2 <- ggplot(iris,aes(x=Sepal.Length,y=Sepal.Width))+  
  geom_point() + geom_smooth()
```

x1|x2



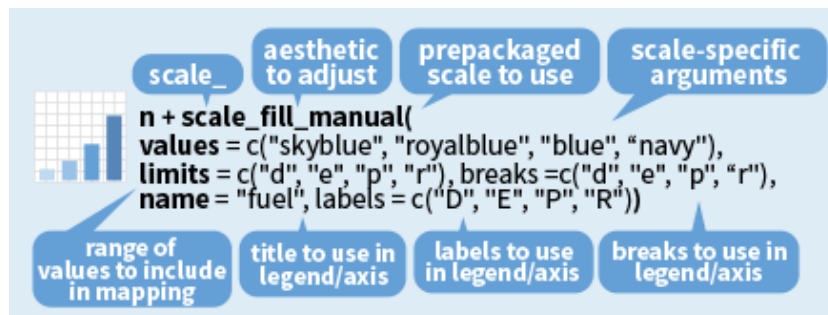
Multiple Geoms

```
ggplot(iris,aes(x=Sepal.Length,y=Sepal.Width))+  
  geom_point()+  
  geom_line()+  
  geom_smooth()+  
  geom_rug()+  
  geom_step()+  
  geom_text(data=subset(iris,iris$Species=="setosa"),aes(label=Species))
```

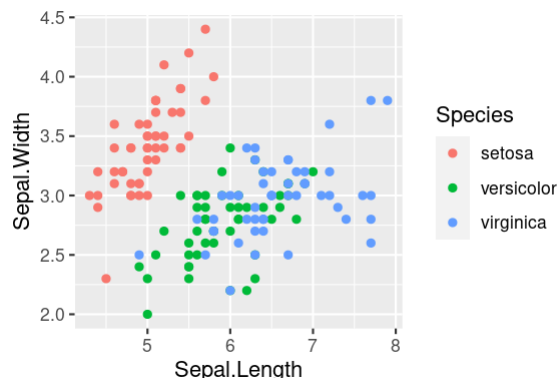


Scales • Discrete Colors

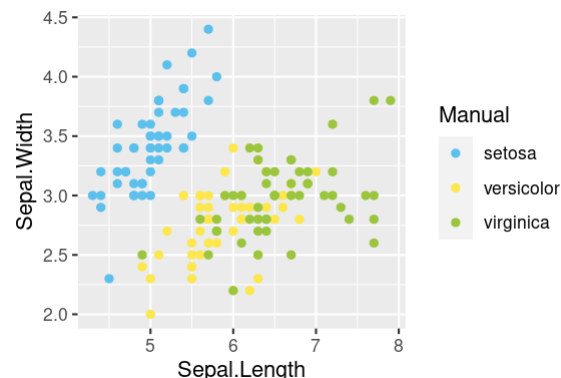
- scales: position, color, fill, size, shape, alpha, linetype
- syntax: `scale_aesthetic>_<type>`



```
p <- ggplot(iris)+geom_point(aes(x=Sepal.Length,
y=Sepal.Width,color=Species))
p
```



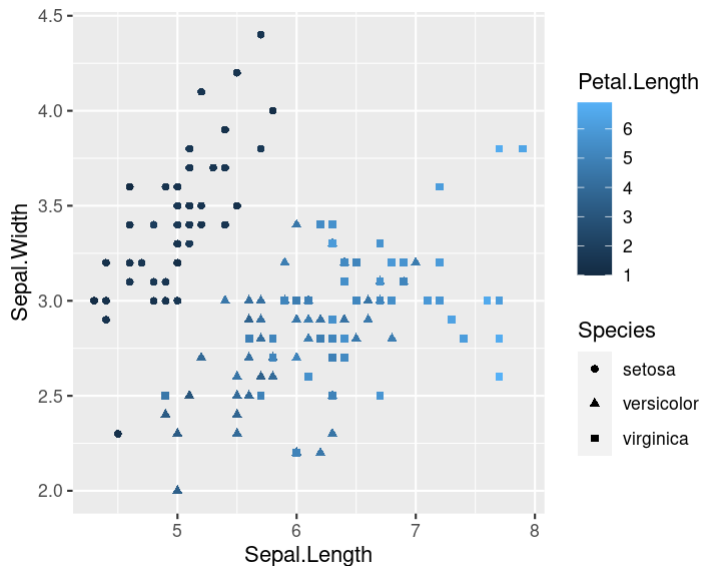
```
p + scale_color_manual(
  name="Manual",
  values=c("#5BC0EB", "#FDE74C", "#9BC53D"))
```



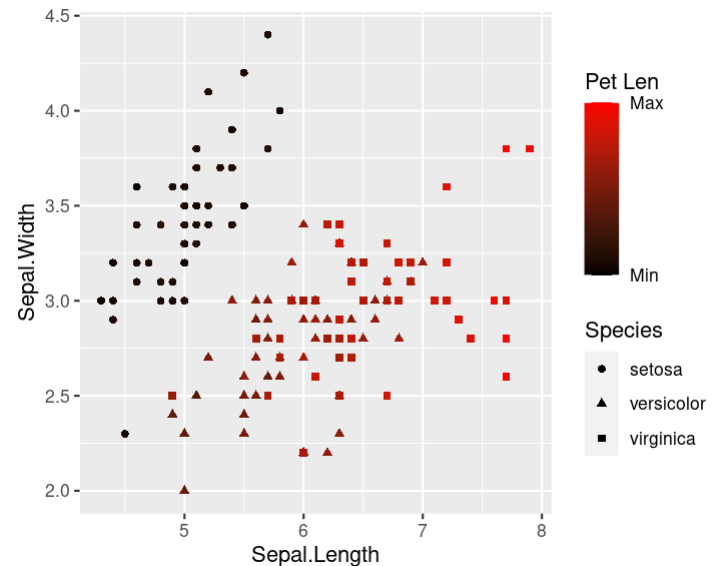
Scales • Continuous Colors

- In RStudio, type `scale_`, then press **TAB**

```
p <- ggplot(iris)+  
  geom_point(aes(x=Sepal.Length,  
                 y=Sepal.Width,  
                 shape=Species,color=Petal.Length))  
p
```



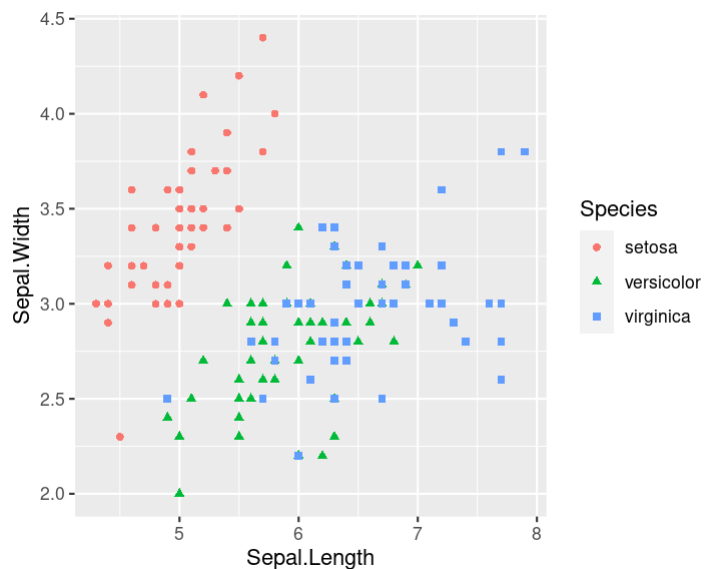
```
p +  
  scale_color_gradient(name="Pet Len",  
                       breaks=range(iris$Petal.Length),  
                       labels=c("Min", "Max"),  
                       low="black",high="red")
```



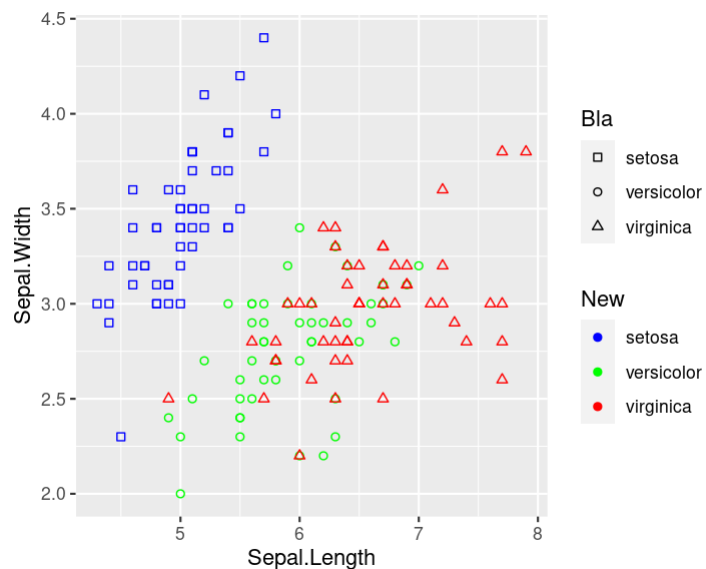
Scales • Shape

```
p <- ggplot(iris)+  
  geom_point(aes(x=Sepal.Length,  
                 y=Sepal.Width,  
                 shape=Species,color=Species))
```

p



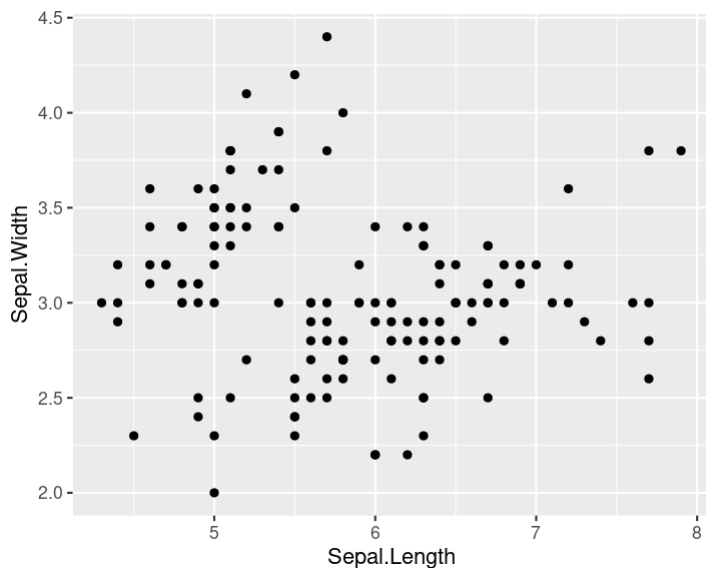
```
p +  
  scale_color_manual(name="New",  
                    values=c("blue","green","red"))+  
  scale_shape_manual(name="Bla",values=c(0,1,2
```



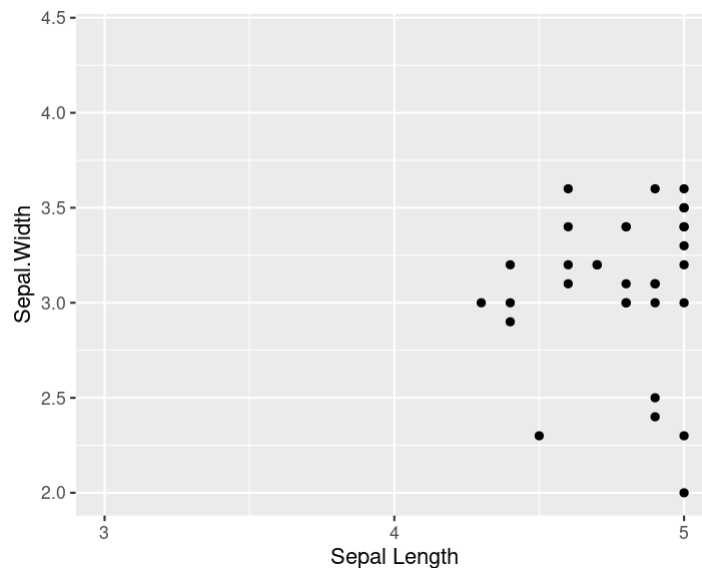
Scales • Axes

- scales: x, y
- syntax: `scale_<axis>_<type>`
- arguments: name, limits, breaks, labels

```
p <- ggplot(iris)+  
  geom_point(aes(x=Sepal.Length,  
                 y=Sepal.Width))  
p
```



```
p + scale_color_manual(name="New",  
                       values=c("blue", "green", "red"))+  
  scale_x_continuous(name="Sepal Length",  
                    breaks=seq(1,8),limits=c(3,5))
```

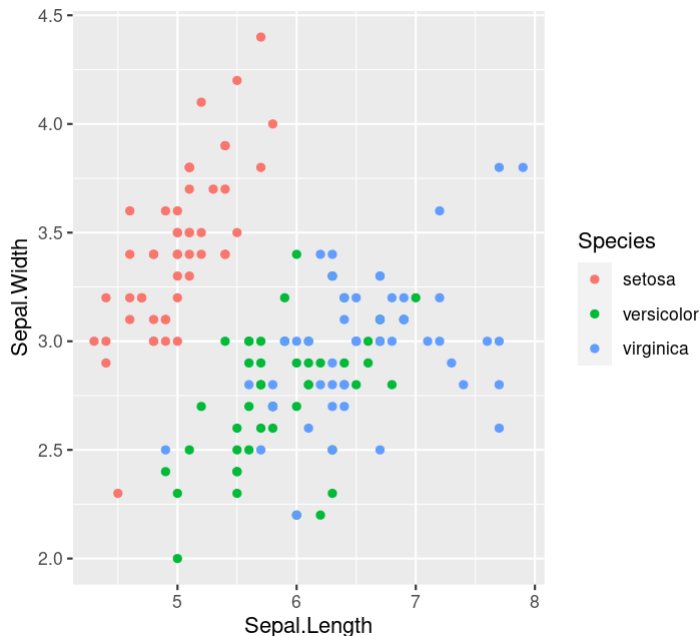


Facets • `facet_wrap`

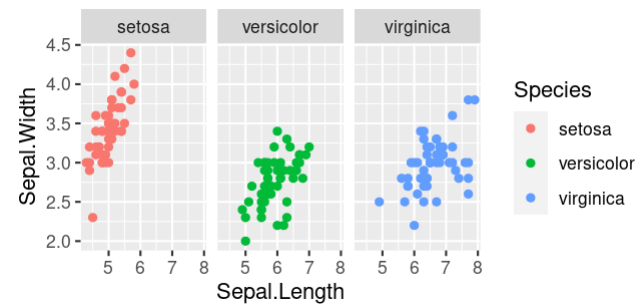
- Split to subplots based on variable(s)
- Facetting in one dimension

```
p <- ggplot(iris)+  
  geom_point(aes(x=Sepal.Length,  
                y=Sepal.Width,  
                color=Species))
```

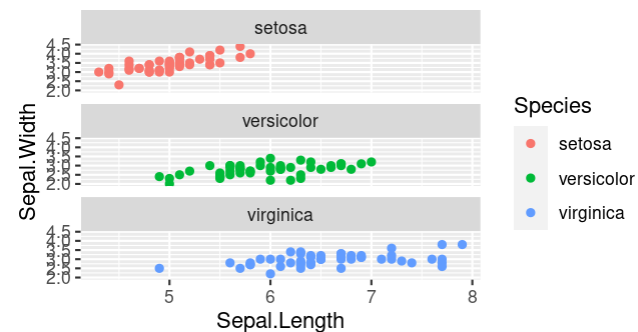
p



```
p + facet_wrap(~Species)
```



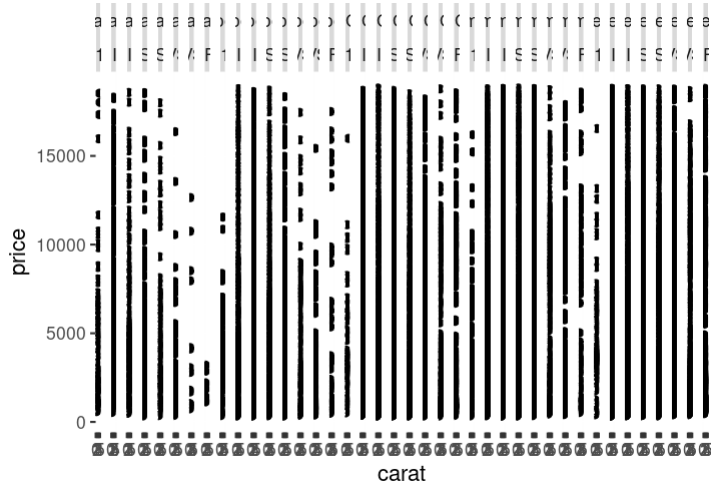
```
p + facet_wrap(~Species, nrow=3)
```



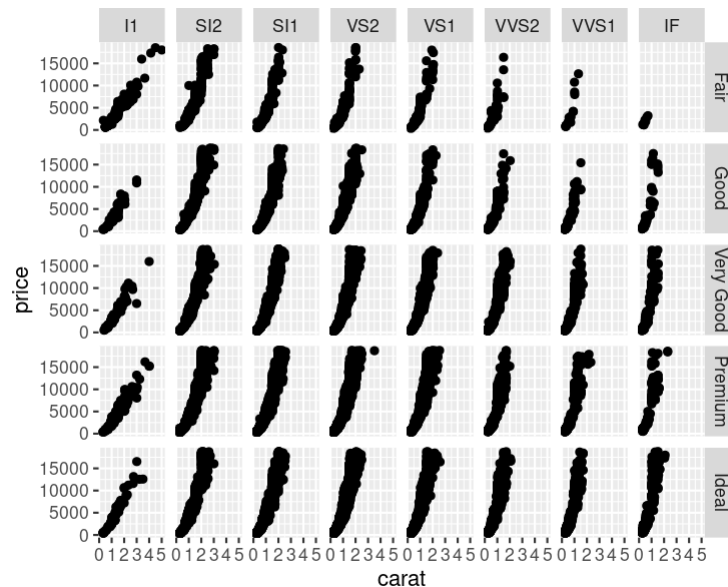
Facets • facet_grid

- Facetting in two dimensions

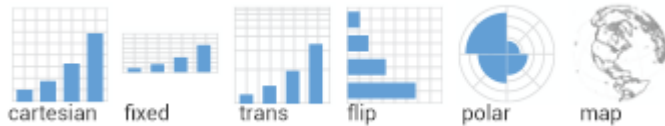
```
p <- diamonds %>%  
  ggplot(aes(carat,price))+  
  geom_point()  
p + facet_grid(~cut+clarity)
```



```
p + facet_grid(cut~clarity)
```

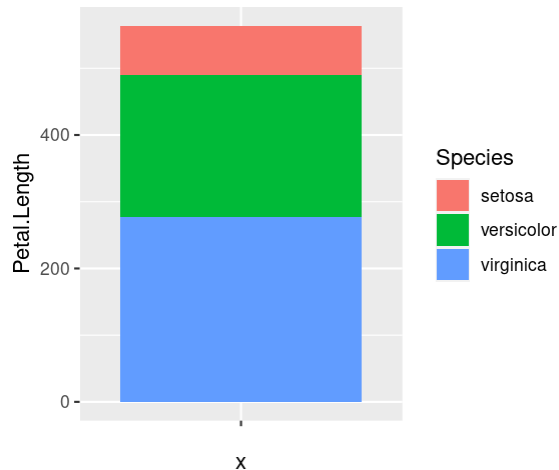


Coordinate Systems

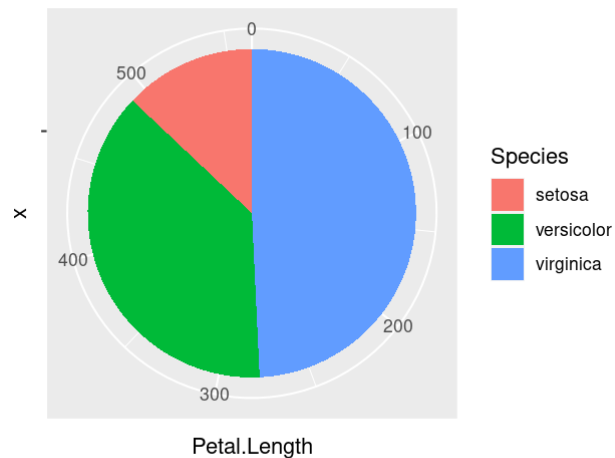


- `coord_cartesian(xlim=c(2,8))` for zooming in
- `coord_map` for controlling limits on maps
- `coord_polar`

```
p <- ggplot(iris,aes(x="",y=Petal.Length,fill=Species))  
  geom_bar(stat="identity")  
p
```



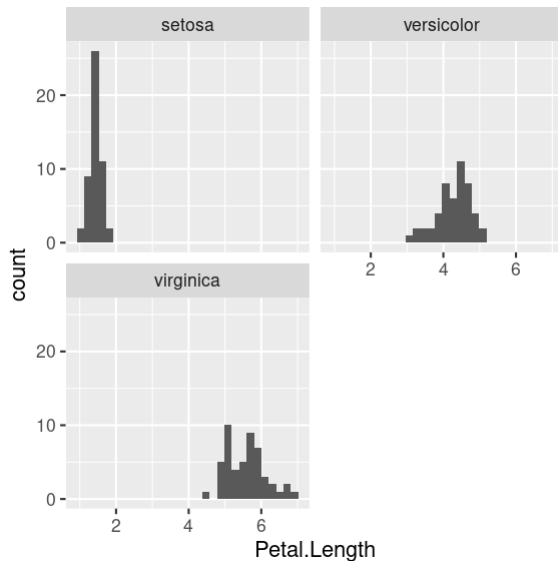
```
p+coord_polar("y",start=0)
```



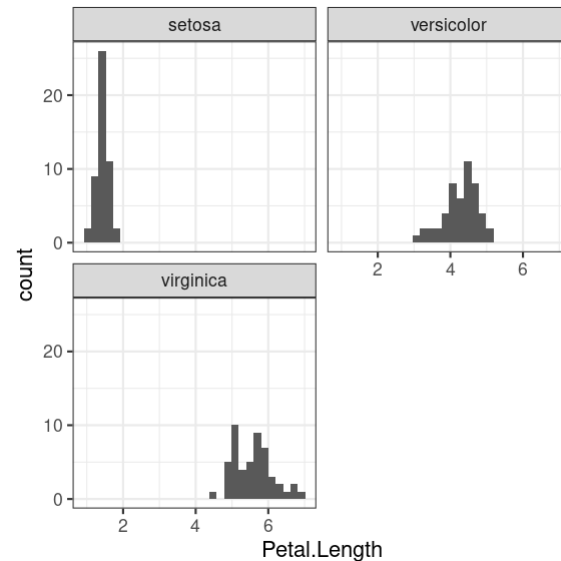
Theme

- Modify non-data plot elements/appearance
- Axis labels, panel colors, legend appearance etc
- Save a particular appearance for reuse
- `?theme`

```
ggplot(iris,aes(Petal.Length))+  
  geom_histogram()+  
  facet_wrap(~Species,nrow=2)+  
  theme_grey()
```



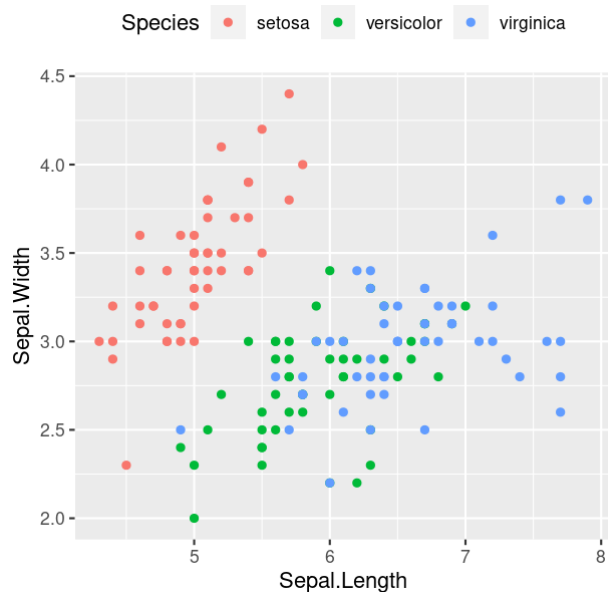
```
ggplot(iris,aes(Petal.Length))+  
  geom_histogram()+  
  facet_wrap(~Species,nrow=2)+  
  theme_bw()
```



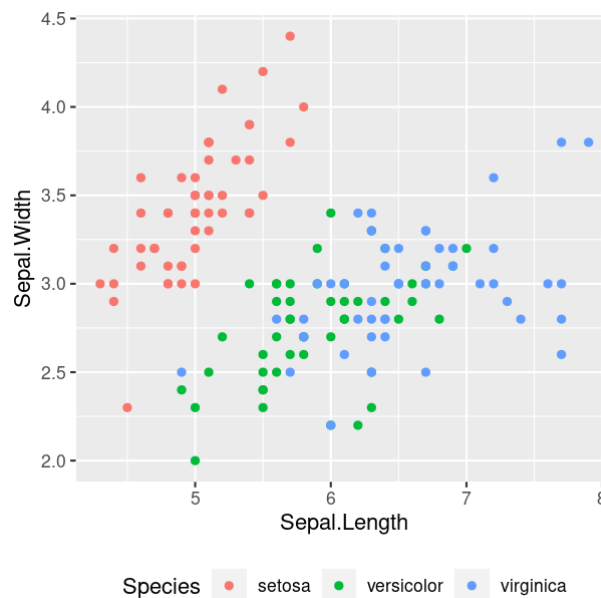
Theme • Legend

```
p <- ggplot(iris)+  
  geom_point(aes(x=Sepal.Length,  
                y=Sepal.Width,  
                color=Species))
```

```
p + theme(legend.position="top")
```



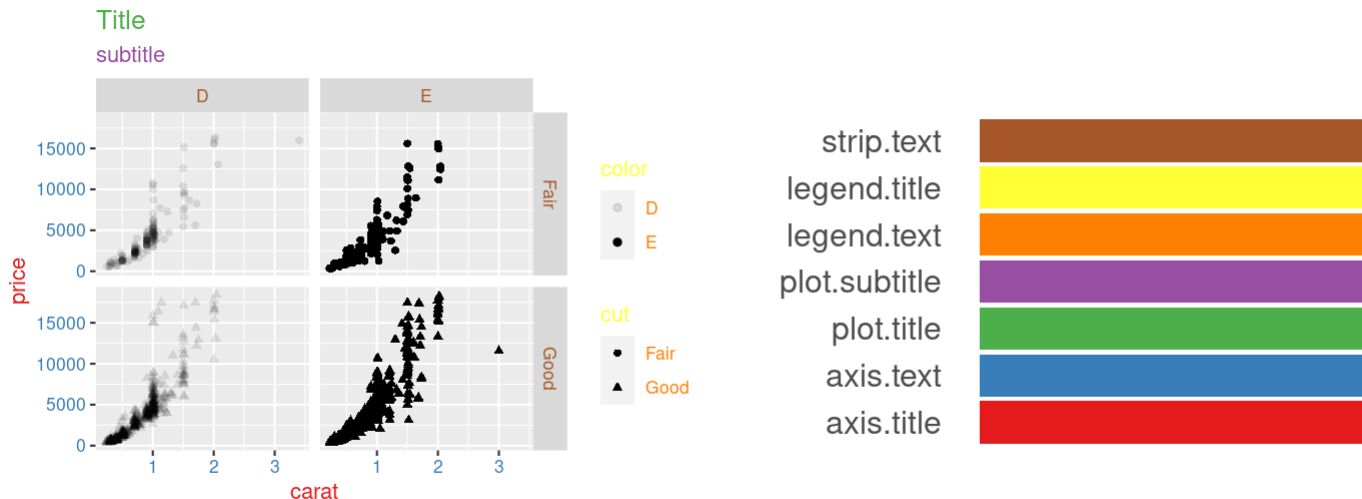
```
p + theme(legend.position="bottom")
```



Theme • Text

```
element_text(family=NULL,face=NULL,color=NULL,size=NULL,hjust=NULL,
             vjust=NULL, angle=NULL,lineheight=NULL,margin = NULL)
```

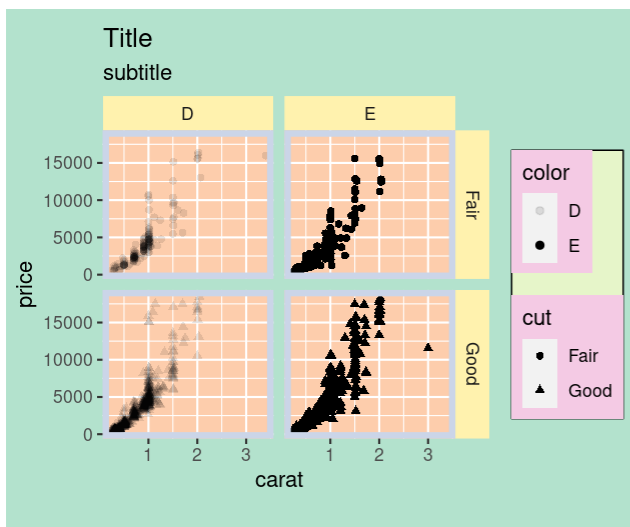
```
p <- p + theme(
  axis.title=element_text(color="#e41a1c"),
  axis.text=element_text(color="#377eb8"),
  plot.title=element_text(color="#4daf4a"),
  plot.subtitle=element_text(color="#984ea3"),
  legend.text=element_text(color="#ff7f00"),
  legend.title=element_text(color="#ffff33"),
  strip.text=element_text(color="#a65628")
)
```



Theme • Rect

```
element_rect(fill=NULL,color=NULL,size=NULL,linetype=NULL)
```

```
p <- p + theme(  
  plot.background=element_rect(fill="#b3e2cd"),  
  panel.background=element_rect(fill="#fcdac"),  
  panel.border=element_rect(fill=NA,color="#cbd5e8",size=3),  
  legend.background=element_rect(fill="#f4cae4"),  
  legend.box.background=element_rect(fill="#e6f5c9"),  
  strip.background=element_rect(fill="#fff2ae")  
)
```



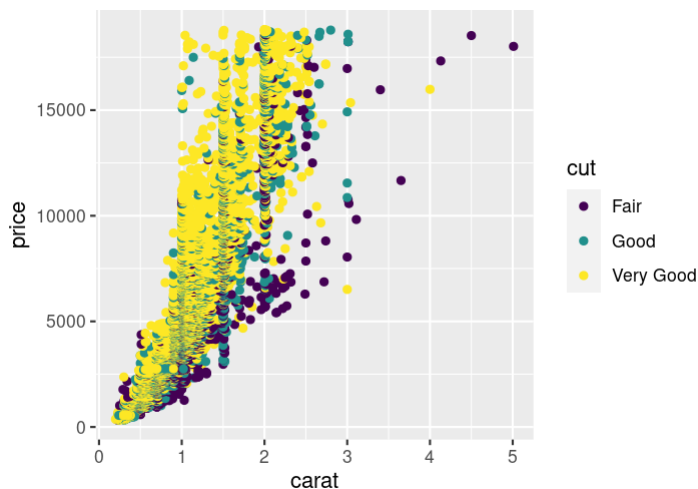
strip.background
legend.box.background
legend.background
panel.border
panel.background
plot.background



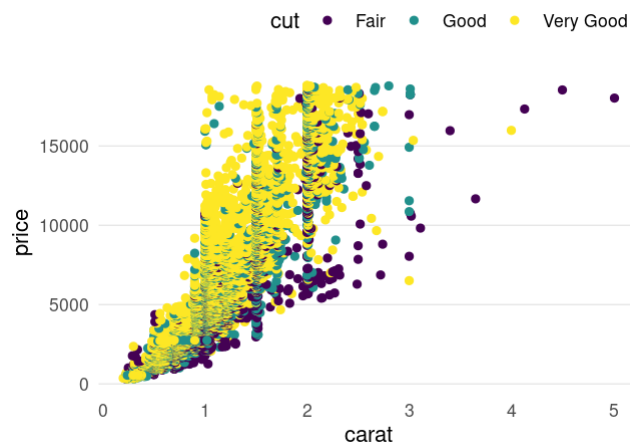
Theme • Reuse

```
newtheme <- theme_bw() + theme(  
  axis.ticks=element_blank(),  
  panel.background=element_rect(fill="white"),  
  panel.grid.minor=element_blank(),  
  panel.grid.major.x=element_blank(),  
  panel.grid.major.y=element_line(size=0.3,color="grey90"),  
  panel.border=element_blank(),  
  legend.position="top",  
  legend.justification="right"  
)
```

p



p + newtheme



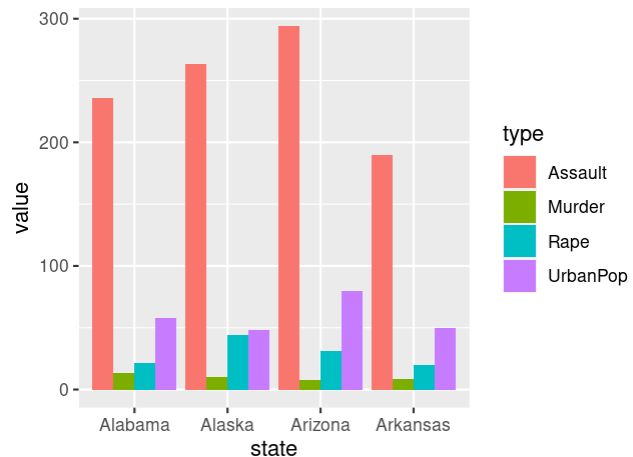
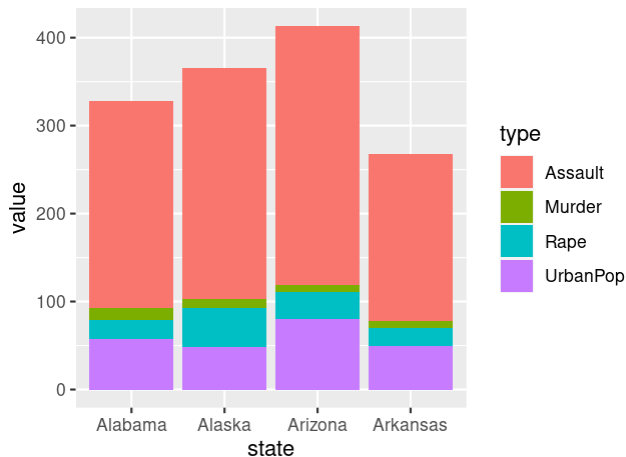
Position

```
##           Murder Assault UrbanPop Rape
## Alabama    13.2     236      58 21.2
## Alaska     10.0     263      48 44.5
## Arizona      8.1     294      80 31.0
```

```
us <- USArrests %>% mutate(state=rownames(.)) %>% slice(1:4) %>%
  gather(key=type,value=value,-state)
p <- ggplot(us,aes(x=state,y=value,fill=type))
```

```
p + geom_bar(stat="identity",position="stack"
```

```
p + geom_bar(stat="identity",position="dodge"
```



Saving plots

```
p <- ggplot(iris,aes(Petal.Length,Sepal.Length,color=Species))+  
  geom_point()
```

- `ggplot2` plots can be saved just like base plots

```
png("plot.png",height=5,width=7,units="cm",res=200)  
print(p)  
dev.off()
```

- `ggplot2` package offers a convenient function

```
ggsave("plot.png",p,height=5,width=7,units="cm",dpi=200,type="cairo")
```

- Use `type="cairo"` for nicer anti-aliasing
- Note that default units in `png` is pixels while in `ggsave` it's inches

Extensions

- **gridExtra**: Extends grid graphics functionality
- **ggpubr**: Useful functions to prepare plots for publication
- **cowplot**: Combining plots
- **ggthemes**: Set of extra themes
- **ggthemr**: More themes
- **ggsci**: Color palettes for scales
- **ggrepel**: Advanced text labels including overlap control
- **ggmap**: Dedicated to mapping
- **ggraph**: Network graphs
- **ggiraph**: Converting ggplot2 to interactive graphics

Thank you. Questions?

Graphics from  freepik.com

Created: 09-Aug-2023 • Roy Francis • SciLifeLab • NBIS

