Object-oriented programming

managing complexity

Learning Aims

- After this session you will
    - understand what object-oriented programming is
    - know how OO can be used to manage complexity
    - understand how OO works in R
    - understand generic functions and how they work
    - be able to make your own R objects

Object orientation

- In R, pretty much everything is an **object**
- Object orientation is a style of programming that breaks down complex functionality into objects that talk to each other
- There are two OO systems built into R:
    - the original and less formal S3 system
    - the newer and more formal S4 system

S3

- S3 objects are lists with class

```
cars.model<-lm(cars$dist~cars$speed)
class(cars.model)
str(cars.model)
```

- They have an agreed structure, but there is no formal contract

Generic functions

- The plot( ) function isn't really magic: it's a generic function
- When you pass plot( ) an object, R checks what type of object it is, and passes responsibility on to one of a range of plot( ) functions

```
methods(plot)
```

S4

- S4 is a more formal system of OO
- Bioconductor has adopted S4 as its preferred method for new packages
- S4 imposes formal class definitions and checks for conformity
- Method dispatch is based on function signatures