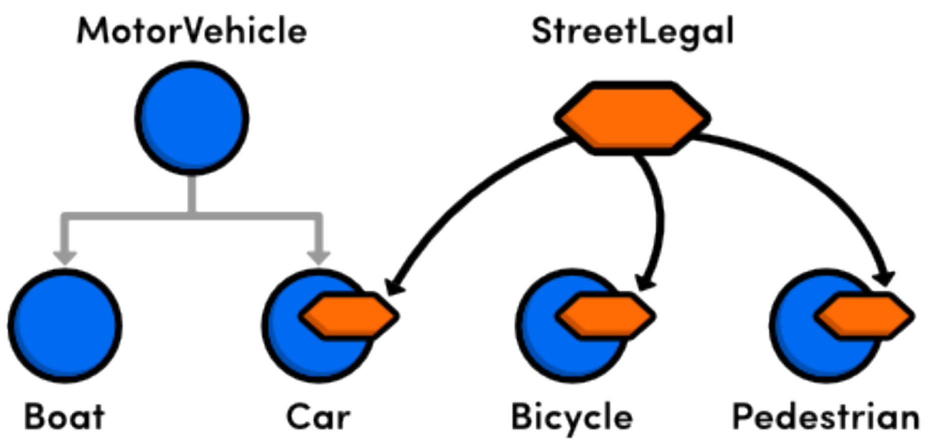
Protocols

A protocol is a group of related properties and methods that can be implemented by any class. They are more flexible than a normal class interface, since they let you reuse a single API declaration in completely unrelated classes. This makes it possible to represent horizontal relationships on top of an existing class hierarchy.



```
// StreetLegal.h
#import <Foundation/Foundation.h>

@protocol StreetLegal <NSObject>

- (void)signalStop;
- (void)signalLeftTurn;
- (void)signalRightTurn;

@end
```

```
// Bicycle.h
#import <Foundation/Foundation.h>
#import "StreetLegal.h"
```

```
@interface Bicycle : NSObject <StreetLegal>

- (void)startPedaling;
- (void)removeFrontWheel;
- (void)lockToStructure:(id)theStructure;

@end
```

```
// Bicycle.m
#import "Bicycle.h"

@implementation Bicycle

- (void)signalStop {
    NSLog(@"Bending left arm downwards");
}
- (void)signalLeftTurn {
    NSLog(@"Extending left arm outwards");
}
- (void)signalRightTurn {
    NSLog(@"Bending left arm upwards");
}
- (void)startPedaling {
    NSLog(@"Here we go!");
}
- (void)removeFrontWheel {
    NSLog(@"Front wheel is off."
          "Should probably replace that before pedaling...");
}
- (void)lockToStructure:(id)theStructure {
    NSLog(@"Locked to structure. Don't forget the combination!");
}

@end
```

```
// main.m
#import <Foundation/Foundation.h>
#import "Bicycle.h"
```

```
int main(int argc, const char * argv[]) {
    @autoreleasepool {
        Bicycle *bike = [[Bicycle alloc] init];
        [bike startPedaling];
        [bike signalLeftTurn];
        [bike signalStop];
        [bike lockToStructure:nil];
    }
    return 0;
}
```

Type Checking With Protocols

Just like classes, protocols can be used to type check variables. To make sure an object adopts a protocol, put the protocol name after the data type in the variable declaration, as shown below. The next code snippet also assumes that you have created a Car class that adopts theStreetLegal protocol:

```
// main.m
#import <Foundation/Foundation.h>
#import "Bicycle.h"
#import "Car.h"
#import "StreetLegal.h"

int main(int argc, const char * argv[]) {
    @autoreleasepool {
        id <StreetLegal> mysteryVehicle = [[Car alloc] init];
        [mysteryVehicle signalLeftTurn];

        mysteryVehicle = [[Bicycle alloc] init];
        [mysteryVehicle signalLeftTurn];
    }
    return 0;
}
```