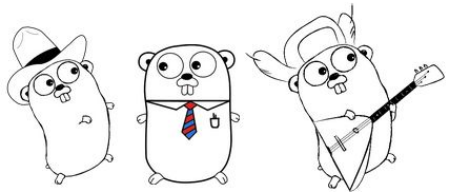
Interesting 8 facts about Go

Brief introduction about Go



- Go (golang) is a programming language created at Google in 2009
- Go is extremely simple language, that’s the reason why it’s getting popular
- You need to have different mindset to write go because Go don’t have most of functionalities which other languages have
- Open source repository which written in Go:
  - Docker
  - Kubernetes (Management tool for multiple Docker images)
  - Hugo (Static site generator)

Hello World!

1. No exceptions

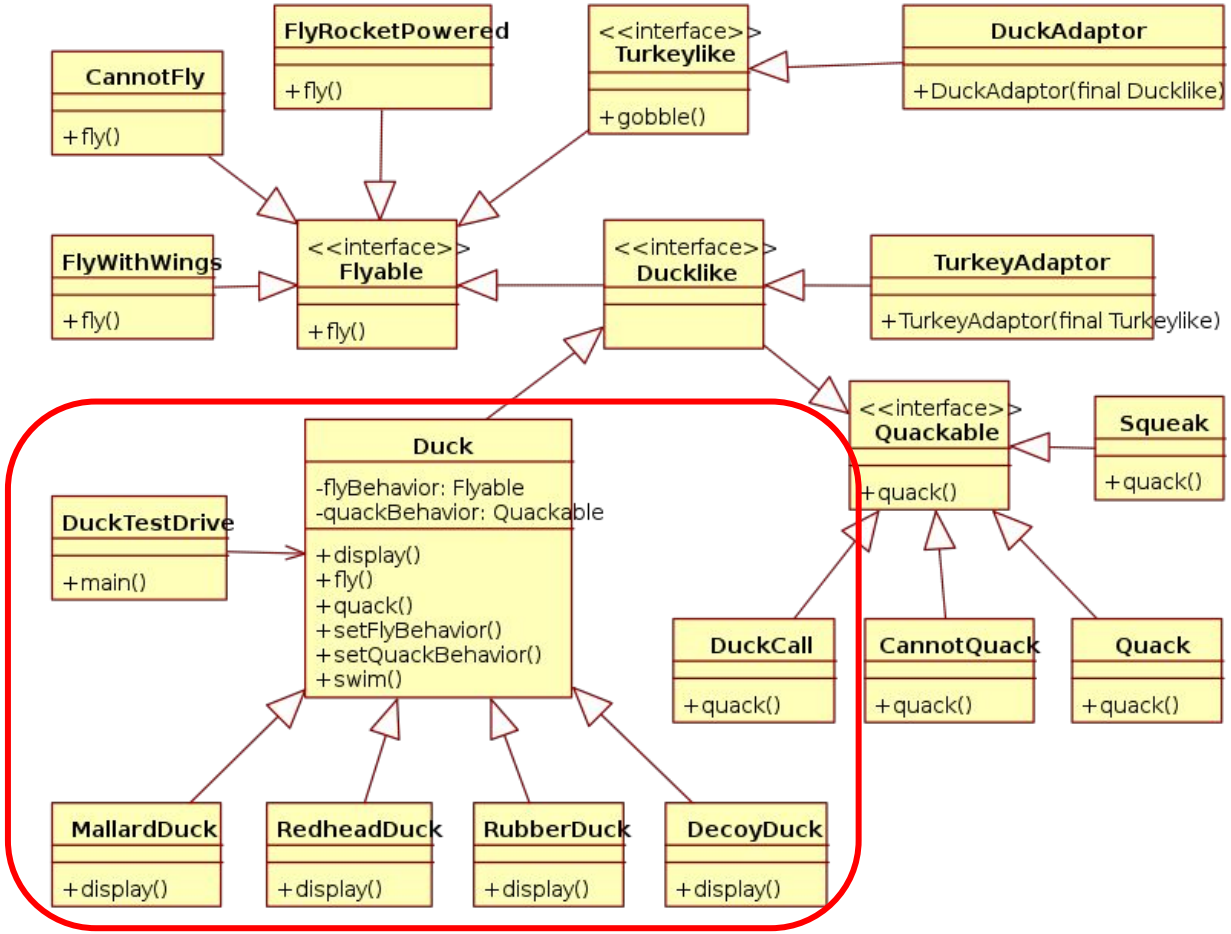
- Go doesn’t have try-catch clause, you must handle it immediately and explicitly
- In Go, functions must return error if error occured
- Code Example(noexceptions/main.go):
  - Input a string from Standard input
  - Convert it to int value
  - Print the value

2. Compiled language

- Go is a compiled language, not interpreted language
- Every time you compile source code, it creates executable file which includes all libraries
- It’s pretty good for writing application for Docker because:
  - No dependency
  - Less image size - don’t need to place libraries on Docker image
- Code Example(docker/main.go and Dockerfile):
  - Go application: Output “Hello World!”
  - Build the application on Golang Docker container
  - Place the executable file on a Linux docker image

3. No Class system

- **Go doesn’t have class system! Neither constructor and destructor**
- Go has C-like struct, that’s good enough to solve OOP problems
- Since it doesn’t have class system, of course it doesn’t have generics and inheritance
- Example(noclass/main.go):
  - Implement Duck class diagram in Go
  - (Class diagram is in the next page)


FlyRocketPowered
+fly()
<<interface>>
Turkeylike
+gobble()
DuckAdaptor
+DuckAdaptor(final Ducklike)
CannotFly
+fly()
FlyWithWings
+fly()
<<interface>>
Flyable
+fly()
<<interface>>
Ducklike
TurkeyAdaptor
+TurkeyAdaptor(final Turkeylike)
<<interface>>
Quackable
+quack()
Squeak
+quack()
Duck
-flyBehavior: Flyable
-quackBehavior: Quackable
+display()
+fly()
+quack()
+setFlyBehavior()
+setQuackBehavior()
+swim()
DuckTestDrive
+main()
DuckCall
+quack()
CannotQuack
+quack()
Quack
+quack()
MallardDuck
+display()
RedheadDuck
+display()
RubberDuck
+display()
DecoyDuck
+display()

4. Interface is important

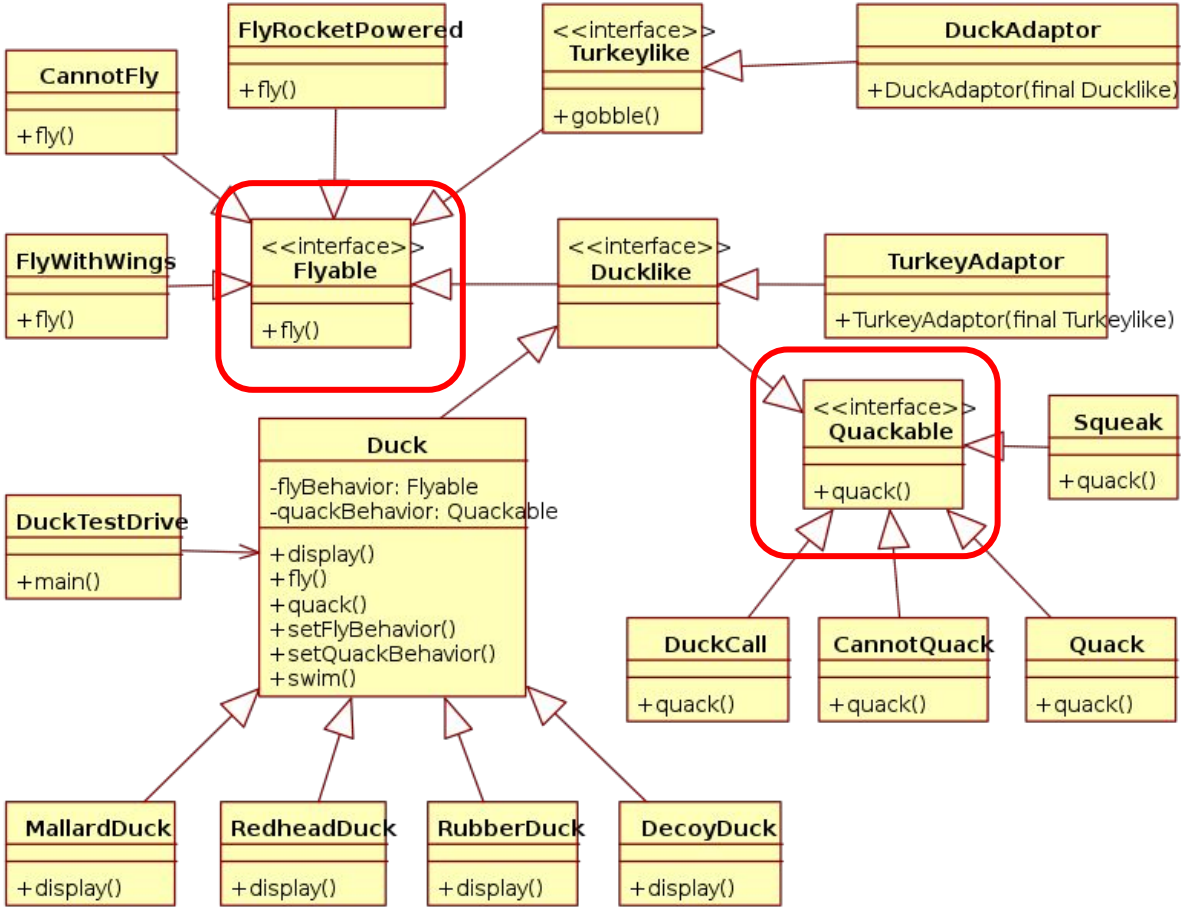
- Go has interface, but it’s place somewhere between Java’s strict interface and duck typing
- Example(gointerface/main.go):
  - Implement Flyable and Quackable interface for Duck class

Strict Typing (Java, C#)

```
class Duck implements IDuck
{
  public void Quack() { ... }
  public void Walk() { ... }
}
class OtherDuck implements IDuck
{
  public void Quack() { ... }
  public void Walk() { ... }
}
...
void M(IDuck bird)
{
  bird.Quack();
  bird.Walk();
}
...
M(new Duck());
M(new OtherDuck());
```

Duck Typing (PHP, Ruby, Python...etc)

```
class Duck
{
  public void Quack() { ... }
  public void Walk() { ... }
}
class OtherDuck
{
  public void Quack() { ... }
  public void Walk() { ... }
}
...
void M(bird)
{
  bird.Quack();
  bird.Walk();
}
...
M(new Duck());
M(new OtherDuck());
```


FlyRocketPowered
+fly()
<<interface>>
Turkeylike
+gobble()
DuckAdaptor
+DuckAdaptor(final Ducklike)
CannotFly
+fly()
<<interface>>
Flyable
+fly()
FlyWithWings
+fly()
<<interface>>
Ducklike
TurkeyAdaptor
+TurkeyAdaptor(final Turkeylike)
<<interface>>
Quackable
+quack()
Squeak
+quack()
Duck
-flyBehavior: Flyable
-quackBehavior: Quackable
+display()
+fly()
+quack()
+setFlyBehavior()
+setQuackBehavior()
+swim()
DuckTestDrive
+main()
DuckCall
+quack()
CannotQuack
+quack()
Quack
+quack()
MallardDuck
+display()
RedheadDuck
+display()
RubberDuck
+display()
DecoyDuck
+display()

5. Not functional programming language

- Go is procedural programming language, not functional one
- No syntax sugar and elegant way to write code like Python and Ruby
- Examples(compare to python):
  - 1. Multiple Each Item in a List by 2 (procedual1/main.go)
    - >>> print map(lambda x: x * 2, range(1,11))
    - [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
  - 2. Sum a List of Numbers (procedual2/main.go)
    - >>> print sum(range(1,1001))
    - 500500

6. Super cool default commands

- Go has extremely useful commands which can be used by editors
- Example:
  - gofmt: Format code automatically following Go coding guideline
  - gorename: Rename function name, variable name with static analysis
  - godoc: Generate go documentation -> example

7. Great performance

- Based on [The Computer Language Benchmarks Game](),
- More than 25 times faster compared to PHP, great memory usage

fasta

source	secs	mem	gz	cpu	cpu load
Go	**2.17**	3,404	1358	5.83	58% 69% 66% 77%
PHP	59.37	8,896	1030	59.36	5% 2% 3% 100%

mandelbrot

source	secs	mem	gz	cpu	cpu load
Go	**5.48**	30,704	905	21.74	100% 100% 99% 100%
PHP	125.17	136,776	863	499.16	100% 100% 100% 100%

8. Real world of Go

- Go is good at to implement some specific applications, but not like Full Stack Web frameworks like Ruby on Rails, Django
- API Servers
  - Now microservice architecture is getting popular, and to have api server is common pattern
  - To implement API server with Go is a good option because of the performance reason
  - Particularly, it has file validation/modification, image related tasks like shrink images, it dramatically improves the performance
- Consumers
  - Messaging queue service like Amazon SQS is quite important to send emails and notifications
  - Go has a great performance with concurrent and parallel programming