Generics

Takeaways

Use the same code on many (potentially unrelated) types:

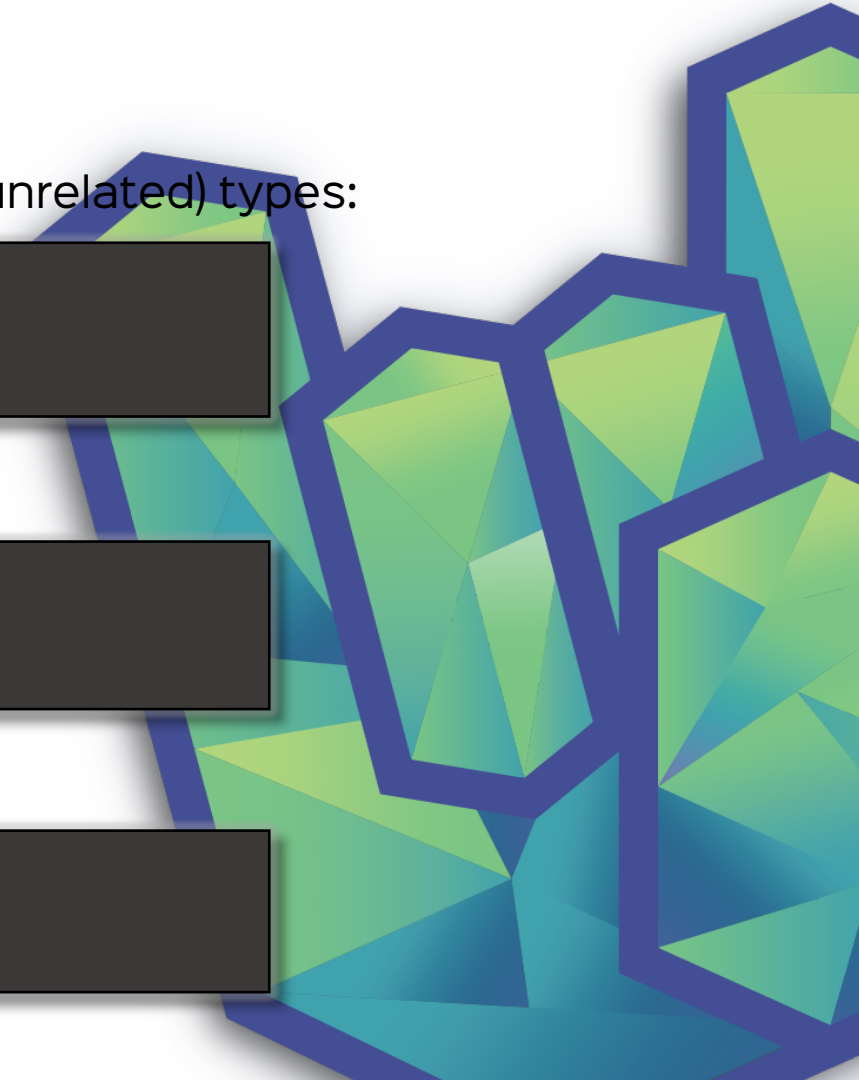
```
trait List[T] {
  def add(elem: T)
}
```

Generic methods

```
object List {
  def single[A](element: A): List[A] = ???
}
```

Multiple type parameters

```
trait Map[Key, Value] {
  ...
}
```



Variance Basics

Variance: if B extends A, *should List[B] extend List[A]?*

```
trait List[+A]
```

yes (covariant)

```
trait List[A]
```

no (invariant) - default

```
trait List[-A]
```

hell no! (contravariant)

Bounded types

```
class Car
class Supercar extends Car
class Garage[T <: Car](car: T)
```

An annoying variance problem

Scala rocks