# J13 Type Inference

Generic type inference
Lambda expressions
Local variables

# Type Inference

The Java compiler can infer many types from context, cutting down on boilerplate code, and simplifying refactoring.

Instantiating generic classes:

```
LinkedList<String> list = new LinkedList();
```

Generic methods:

```
public <T> void add(T value) { }

list.add("A String");
```

# Local Variables

With the `var` keyword, Java can infer the type of a local variable from its initialization expression.

The most specific type is inferred.

```
var theAnswer = 42;
var bike = new Bike();
var mystery; // invalid – no initializer
var nothing = null; // invalid – too vague
```

# Lambda Expressions

Types of **parameters** to lambda expressions:

```
Predicate<String> nonEmpty = x -> x.length() > 0;
```

However, **can't infer** the type of a lambda expression as a local variable:

```
var lambda = x -> x + 1; // invalid – what type is x?
var lambda = (int x) -> x + 1; // invalid – what is lambda?
IntFunction<Integer> lambda = x -> x + 1; // OK
```

Passing a lambda expression directly to a method normally works, as the method parameter provides the type information.