Supplement: javadoc Comments

For Introduction to Java Programming
By Y. Daniel Liang

1 Introduction

Java supports comments of a special type, referred to as *javadoc comments*. javadoc comments begin with **/**** and end with ***/**. You can use javadoc comments to describe a class, an interface, data fields, and methods. The javadoc comments can be extracted into an HTML file using the JDK's **javadoc** command.

2 An Example

Listing 1 gives an example of a program with javadoc comments.

Listing 1 Loan.java

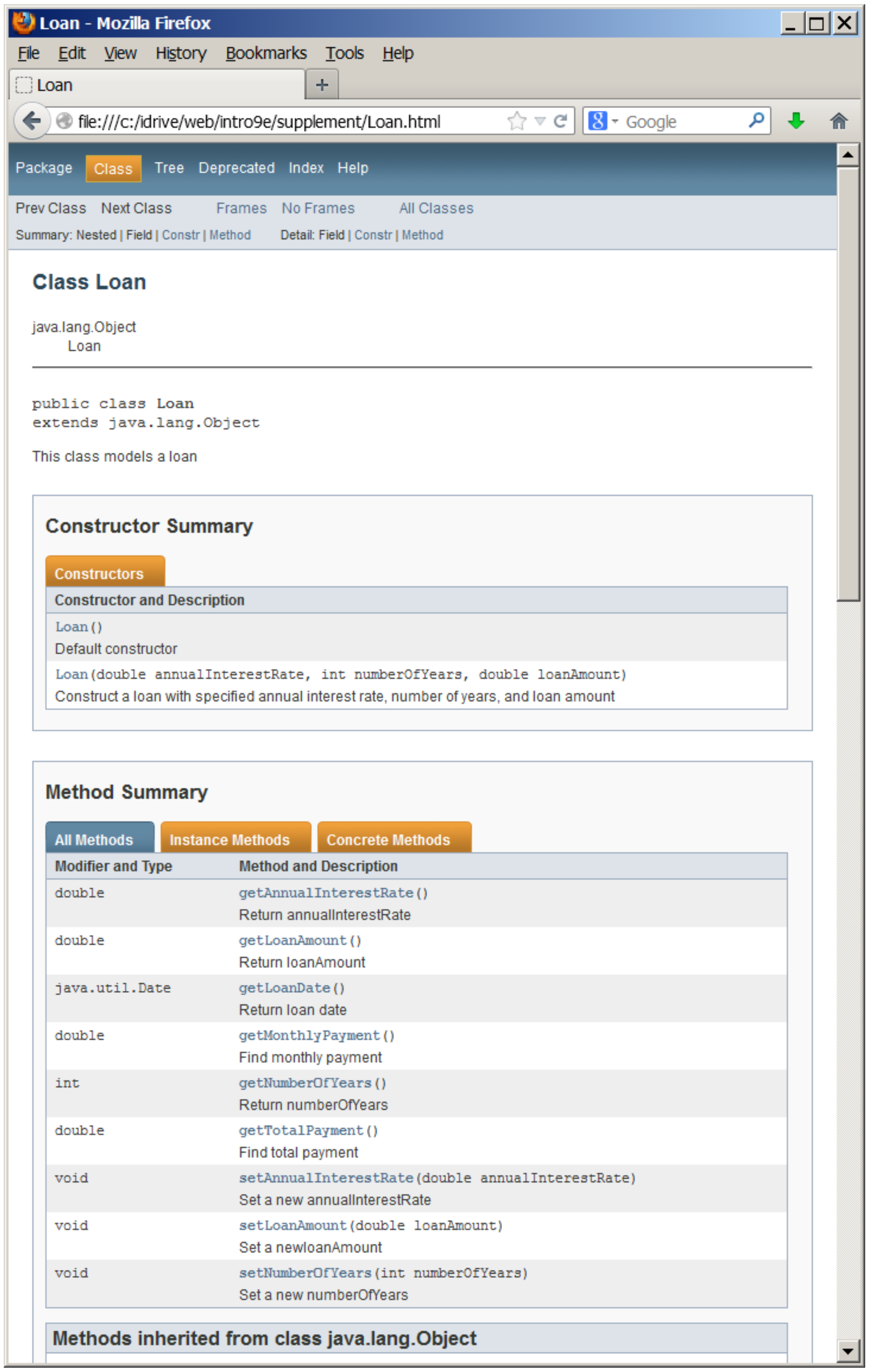
```
/** This class models a loan */

public class Loan {

  /** Data field: annual interest rate */

  private double annualInterestRate;


  /** Data field: number of years */

  private int numberOfYears;


  /** Data field: loan amount */

  private double loanAmount;


  /** Data field: loan creation date */

  private java.util.Date loanDate;


  /** Default constructor */

  public Loan() {

    this(2.5, 1, 1000);

  }


  /** Construct a loan with specified annual interest rate,
    number of years, and loan amount
  */
public Loan(double annualInterestRate, int numberOfYears,
    double loanAmount) {
  this.annualInterestRate = annualInterestRate;
  this.numberOfYears = numberOfYears;
  this.loanAmount = loanAmount;
  loanDate = new java.util.Date();
}

/** Return annualInterestRate */
public double getAnnualInterestRate() {
  return annualInterestRate;
}

/** Set a new annualInterestRate */
public void setAnnualInterestRate(double annualInterestRate) {
  this.annualInterestRate = annualInterestRate;
}

/** Return numberOfYears */
public int getNumberOfYears() {
  return numberOfYears;
}

/** Set a new numberOfYears */
public void setNumberOfYears(int numberOfYears) {
  this.numberOfYears = numberOfYears;
}

/** Return loanAmount */
public double getLoanAmount() {
    return loanAmount;
  }

  /** Set a newloanAmount */
  public void setLoanAmount(double loanAmount) {
    this.loanAmount = loanAmount;
  }

  /** Find monthly payment */
  public double getMonthlyPayment() {
    double monthlyInterestRate = annualInterestRate / 1200;
    double monthlyPayment = loanAmount * monthlyInterestRate / (1 -
      (1 / Math.pow(1 + monthlyInterestRate, numberOfYears * 12)));
    return monthlyPayment;
  }

  /** Find total payment */
  public double getTotalPayment() {
    double totalPayment = getMonthlyPayment() * numberOfYears * 12;
    return totalPayment;
  }

  /** Return loan date */
  public java.util.Date getLoanDate() {
    return loanDate;
  }
}
```

3 Generating HTML Document

You can generate HTML document for the preceding program using the javadoc comment as follows:

javadoc Loan.java

This command processes the source code file Loan.java to generate Loan.html and its supporting HTML files. You can view Loan.html as shown in Figure 1.


Loan - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Loan
file:///c:/idrive/web/intro9e/supplement/Loan.html
Google
Package Class Tree Deprecated Index Help
Prev Class Next Class Frames No Frames All Classes
Summary: Nested | Field | Constr | Method Detail: Field | Constr | Method

Class Loan

java.lang.Object
    Loan

```
public class Loan
extends java.lang.Object
```

This class models a loan

Constructor Summary

Constructors

Constructor and Description
`Loan()` Default constructor
`Loan(double annualInterestRate, int numberOfYears, double loanAmount)` Construct a loan with specified annual interest rate, number of years, and loan amount

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method and Description
`double`	`getAnnualInterestRate()` Return annualInterestRate
`double`	`getLoanAmount()` Return loanAmount
`java.util.Date`	`getLoanDate()` Return loan date
`double`	`getMonthlyPayment()` Find monthly payment
`int`	`getNumberOfYears()` Return numberOfYears
`double`	`getTotalPayment()` Find total payment
`void`	`setAnnualInterestRate(double annualInterestRate)` Set a new annualInterestRate
`void`	`setLoanAmount(double loanAmount)` Set a newloanAmount
`void`	`setNumberOfYears(int numberOfYears)` Set a new numberOfYears

Methods inherited from class java.lang.Object

Figure 1 Loan.html is displayed in a browser.

3 javadoc Tags

You can use javadoc tags to specify the type of the information described in the comments. The commonly used tags are the following:

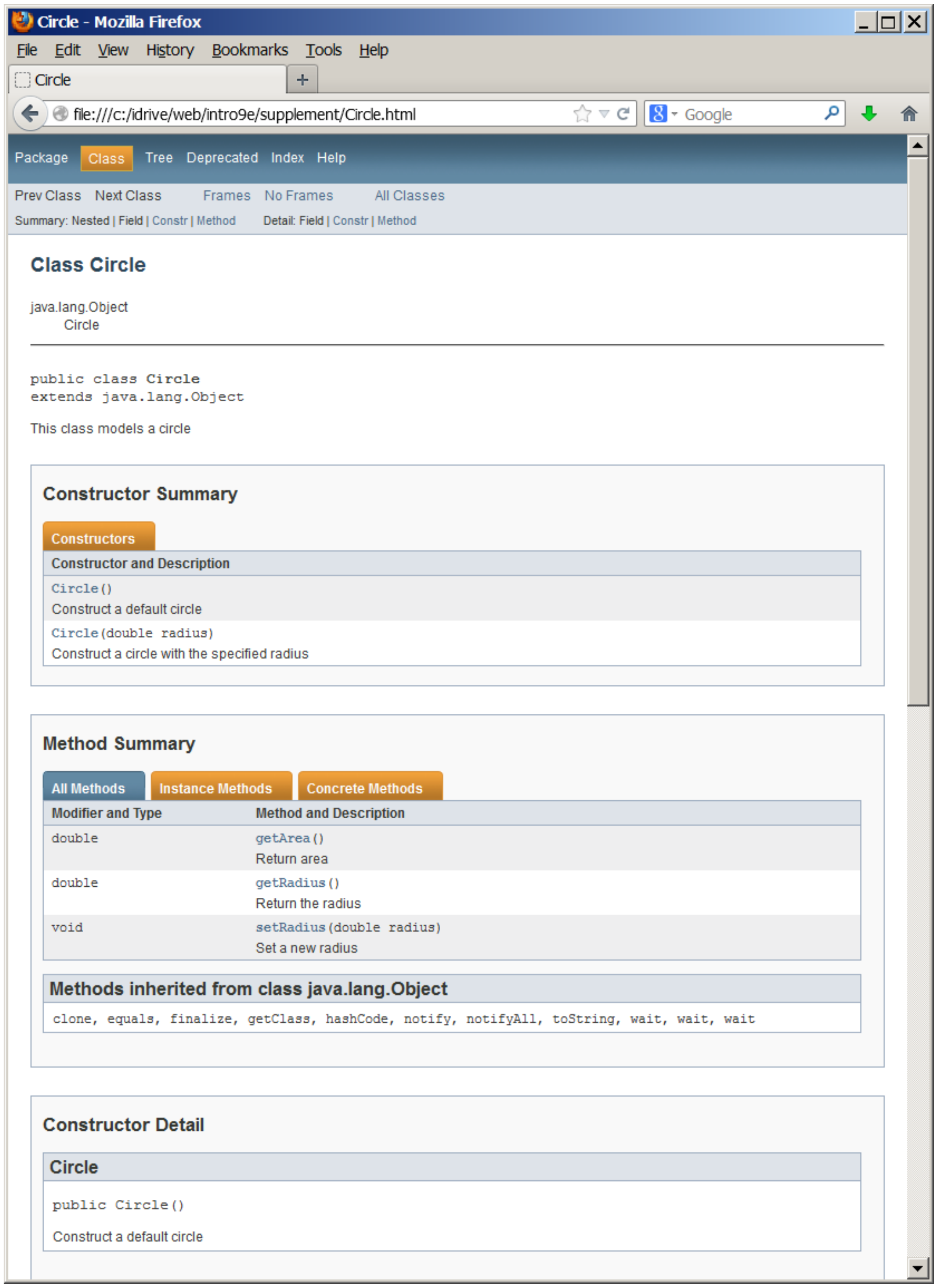
- @author [author name]: identifies the author(s) of a class or interface.
- @version [version]: gives the version of a class or interface.
- @param [parameter name] [parameter description]: describes the parameters in a method or constructor.
- @return [description of return]: describes a return value from a method.
- @exception [exception thrown] [exception description]: describes exception thrown from a method or a constructor.
- @exception [exception thrown] [exception description]: same as @exception

Listing 2 gives an example of using these tags.

Listing 2 Circle.java

```
/** This class models a circle
  *
  * @author Daniel Liang
  * @version 2.1
  */
public class Circle {
  /** Data field: the radius of a circle */
  private double radius;

  /** Construct a default circle */
  public Circle() {
  }

  /** Construct a circle with the specified radius
    * @param radius the radius of the circle
    */
  public Circle(double radius) {
    this.radius = radius;
  }
  /** Return the radius
    * @return radius
    */
  public double getRadius() {
    return radius;
  }

  /** Set a new radius
    * @param radius a new radius
    * @throws IllegalArgumentException if the radius is negative
    */
  public void setRadius(double radius) {
    if (radius < 0)
      throw new IllegalArgumentException("Radius is negative");

    this.radius = radius;
  }

  /** Return area
    * @return the area of the circle
    */
  public double getArea() {
    return radius * radius * Math.PI;
  }
}
```

Figure 2 shows the HTML file generated from the javadoc comments in Circle.java.



Circle - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Circle
file:///c:/idrive/web/intro9e/supplement/Circle.html
Google
Package Class Tree Deprecated Index Help
Prev Class Next Class Frames No Frames All Classes
Summary: Nested | Field | Constr | Method Detail: Field | Constr | Method

Class Circle

java.lang.Object
Circle

```
public class Circle
extends java.lang.Object
```

This class models a circle

Constructor Summary

Constructors

Constructor and Description
`Circle()` Construct a default circle
`Circle(double radius)` Construct a circle with the specified radius

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method and Description
`double`	`getArea()` Return area
`double`	`getRadius()` Return the radius
`void`	`setRadius(double radius)` Set a new radius

Methods inherited from class java.lang.Object

`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

Circle

```
public Circle()
```

Construct a default circle

Figure 2 Circle.html is displayed in a browser.