

Creating and Using Classes



Brice Wilson

Overview



What is a class?

Similarity to classes in other languages

Class members

- **Constructors**
- **Properties**
- **Methods**

Inheritance

Abstract classes

Class expressions



What is a class?

Template for creating objects

Provides state storage and behavior

Encapsulates reusable functionality



Sound Familiar?

Define Types

Properties and
Methods

Constructors

Access Modifiers

Inheritance

Abstract Classes



```
class ReferenceItem {  
    ➔ constructor(title: string, publisher?: string) {  
        // perform initialization here ↑  
    }  
}  
let encyclopedia = new ReferenceItem('WorldPedia', 'WorldPub');
```

Constructors

Method named “constructor” – maximum of one per class

Use optional parameters to call different ways

Executed by using the “new” keyword



Properties and Methods

```
class ReferenceItem {  
    numberOfPages: number;  
    get editor(): string {  
        // Custom getter logic goes here, should return a value  
    }  
    set editor(newEditor: string) {  
        // Custom setter logic goes here  
    }  
    printChapterTitle(chapterNum: number): void {  
        // Print title here  
    }  
}
```



Parameter Properties

```
class Author {  
    name: string;  
    constructor(authorName: string) {  
        name = authorName;  
    }  
}  
  
class Author {  
    constructor(public name: string) { }  
}
```



Static Properties

```
class Library {  
    constructor(public name: string) { }  
    ➔ static description: string = 'A source of knowledge.';  
}  
  
let lib = new Library('New York Public Library');  
let name = lib.name; // available on instances of the class  
let desc = Library.description; // available on the class
```



Access Modifiers

Public

Private

- #

Protected



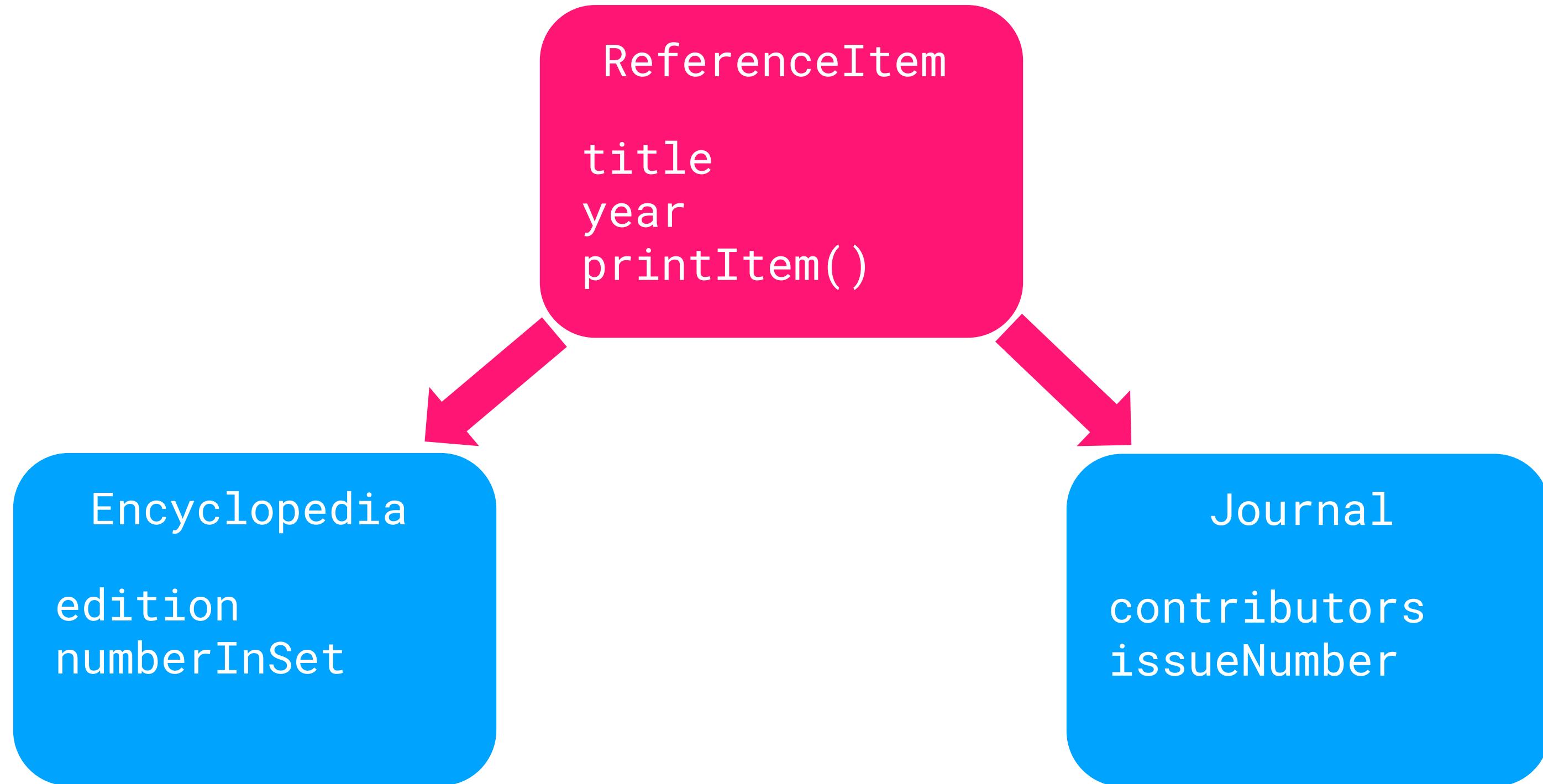
Demo



Creating and using classes



Inheritance



Extending Classes with Inheritance

```
class ReferenceItem {  
    title: string;  
    printItem(): void { // print something here }  
}  
  
class Journal extends ReferenceItem {  
    constructor() {  
        super();  
    }  
    contributors: string[];  
}
```



Demo



Defining an inheritance hierarchy



Abstract Classes

- Created with the “abstract” keyword**
- Base classes that may not be instantiated**
- May contain implementation details**
- Abstract methods are not implemented**



Demo



Creating abstract classes



Demo



Using class expressions



Summary



Classes defined

Class Members

- Constructors
- Properties
- Methods
- Accessors

Inheritance

Abstract Classes

Class Expressions



Up Next:

Organizing Your Code with Modules

