

Un'introduzione a Kitten

Fausto Spoto

16 gennaio 2006

Un linguaggio didattico

- **Una semplificazione di Java**
- Classi, campi, costruttori, metodi
- Ereditarietà, overriding
- Classi diverse in file separati

Un linguaggio didattico

- Una semplificazione di Java
- Classi, campi, costruttori, metodi
- Ereditarietà, overriding
- Classi diverse in file separati

Un linguaggio didattico

- Una semplificazione di Java
- Classi, campi, costruttori, metodi
- Ereditarietà, overriding
- Classi diverse in file separati

Un linguaggio didattico

- Una semplificazione di Java
- Classi, campi, costruttori, metodi
- Ereditarietà, overriding
- Classi diverse in file separati

- Java deve essere installato
- Scaricate `kitten_XXX.tgz` dal sito del corso
- Salvatelo nel punto in cui volete creare la directory `kitten_XXX`
- `tar -zxf kitten_XXX.tgz`
- `cd kitten_XXX`
- `make generatejb`
- Provatelo:
 - `cd testcases`
 - `java -cp .. Kitten Miao.kit`
 - `java Miao`
 - Dovrebbe stampare miao

- Java deve essere installato
- Scaricate `kitten_XXX.tgz` dal sito del corso
- Salvatelo nel punto in cui volete creare la directory `kitten_XXX`
- `tar -zxf kitten_XXX.tgz`
- `cd kitten_XXX`
- `make generatejb`
- Provatelo:
 - `cd testcases`
 - `java -cp .. Kitten Miao.kit`
 - `java Miao`
 - Dovrebbe stampare miao

- Java deve essere installato
- Scaricate `kitten_XXX.tgz` dal sito del corso
- Salvatelo nel punto in cui volete creare la directory `kitten_XXX`
- `tar -zxf kitten_XXX.tgz`
- `cd kitten_XXX`
- `make generatejb`
- Provatelo:
 - `cd testcases`
 - `java -cp .. Kitten Miao.kit`
 - `java Miao`
 - Dovrebbe stampare miao

- Java deve essere installato
- Scaricate `kitten_XXX.tgz` dal sito del corso
- Salvatelo nel punto in cui volete creare la directory `kitten_XXX`
- `tar -zxf kitten_XXX.tgz`
- `cd kitten_XXX`
- `make generatejb`
- Provatelo:
 - `cd testcases`
 - `java -cp .. Kitten Miao.kit`
 - `java Miao`
 - Dovrebbe stampare miao

- Java deve essere installato
- Scaricate `kitten_XXX.tgz` dal sito del corso
- Salvatelo nel punto in cui volete creare la directory `kitten_XXX`
- `tar -zxf kitten_XXX.tgz`
- `cd kitten_XXX`
- `make generatejb`
- Provatelo:
 - `cd testcases`
 - `java -cp .. Kitten Miao.kit`
 - `java Miao`
 - Dovrebbe stampare miao

- Java deve essere installato
- Scaricate `kitten_XXX.tgz` dal sito del corso
- Salvatelo nel punto in cui volete creare la directory `kitten_XXX`
- `tar -zxf kitten_XXX.tgz`
- `cd kitten_XXX`
- `make generatejb`
- Provatelo:
 - `cd testcases`
 - `java -cp .. Kitten Miao.kit`
 - `java Miao`
 - Dovrebbe stampare miao

- Java deve essere installato
- Scaricate `kitten_XXX.tgz` dal sito del corso
- Salvatelo nel punto in cui volete creare la directory `kitten_XXX`
- `tar -zxf kitten_XXX.tgz`
- `cd kitten_XXX`
- `make generatejb`
- Provatelo:
 - `cd testcases`
 - `java -cp .. Kitten Miao.kit`
 - `java Miao`
 - Dovrebbe stampare `miao`

- Java deve essere installato
- Scaricate `kitten_XXX.tgz` dal sito del corso
- Salvatelo nel punto in cui volete creare la directory `kitten_XXX`
- `tar -zxf kitten_XXX.tgz`
- `cd kitten_XXX`
- `make generatejb`
- Provatelo:
 - `cd testcases`
 - `java -cp .. Kitten Miao.kit`
 - `java Miao`
 - Dovrebbe stampare `miao`

- Java deve essere installato
- Scaricate `kitten_XXX.tgz` dal sito del corso
- Salvatelo nel punto in cui volete creare la directory `kitten_XXX`
- `tar -zxf kitten_XXX.tgz`
- `cd kitten_XXX`
- `make generatejb`
- Provatelo:
 - `cd testcases`
 - `java -cp .. Kitten Miao.kit`
 - `java Miao`
 - Dovrebbe stampare `miao`

- Java deve essere installato
- Scaricate `kitten_XXX.tgz` dal sito del corso
- Salvatelo nel punto in cui volete creare la directory `kitten_XXX`
- `tar -zxf kitten_XXX.tgz`
- `cd kitten_XXX`
- `make generatejb`
- Provatelo:
 - `cd testcases`
 - `java -cp .. Kitten Miao.kit`
 - `java Miao`
 - Dovrebbe stampare `miao`

- Java deve essere installato
- Scaricate `kitten_XXX.tgz` dal sito del corso
- Salvatelo nel punto in cui volete creare la directory `kitten_XXX`
- `tar -zxf kitten_XXX.tgz`
- `cd kitten_XXX`
- `make generatejb`
- Provatelo:
 - `cd testcases`
 - `java -cp .. Kitten Miao.kit`
 - `java Miao`
 - Dovrebbe stampare `miao`

- Serve se si vogliono visualizzare alcune strutture dati intermedie della compilazione
- È **vivamente** consigliato installarlo
- Dovrebbe essere già installato nei laboratori didattici: provate a digitare `dot` per vedere se esiste
- È comunque scaricabile dalla pagina del corso

- Serve se si vogliono visualizzare alcune strutture dati intermedie della compilazione
- È **vivamente** consigliato installarlo
- Dovrebbe essere già installato nei laboratori didattici: provate a digitare `dot` per vedere se esiste
- È comunque scaricabile dalla pagina del corso

Il package graphviz

- Serve se si vogliono visualizzare alcune strutture dati intermedie della compilazione
- È **vivamente** consigliato installarlo
- Dovrebbe essere già installato nei laboratori didattici: provate a digitare `dot` per vedere se esiste
- È comunque scaricabile dalla pagina del corso

- Serve se si vogliono visualizzare alcune strutture dati intermedie della compilazione
- È **vivamente** consigliato installarlo
- Dovrebbe essere già installato nei laboratori didattici: provate a digitare `dot` per vedere se esiste
- È comunque scaricabile dalla pagina del corso

Il mio primo programma Kitten

```
class Miao {  
    constructor() {}  
  
    method void main() {  
        "miao\n".output()  
    }  
}
```

- Deve essere contenuto nel file `Miao.kit`
- Una classe eseguibile deve definire il `main` e il costruttore senza parametri
- Se `miao.kit` sta dentro `testcases` (consigliato) allora dovete entrare in `testcases` per compilarlo:

```
cd testcases; java -cp .. Kitten Miao.kit
```
- Si può comunque utilizzare la macro `./compile Miao.kit` dalla directory `kitten_XXX`

Il mio primo programma Kitten

```
class Miao {  
    constructor() {}  
  
    method void main() {  
        "miao\n".output()  
    }  
}
```

- **Deve** essere contenuto nel file `Miao.kit`
- Una classe eseguibile deve definire il `main` e il costruttore senza parametri
- Se `Miao.kit` sta dentro `testcases` (consigliato) allora dovete entrare in `testcases` per compilarlo:

```
cd testcases; java -cp .. Kitten Miao.kit
```
- Si può comunque utilizzare la macro `./compile Miao.kit` dalla directory `kitten_XXX`

Il mio primo programma Kitten

```
class Miao {
    constructor() {}

    method void main() {
        "miao\n".output()
    }
}
```

- **Deve** essere contenuto nel file `Miao.kit`
- Una classe eseguibile deve definire il `main` e il costruttore senza parametri
- Se `Miao.kit` sta dentro `testcases` (consigliato) allora dovete entrare in `testcases` per compilarlo:
`cd testcases; java -cp .. Kitten Miao.kit`
- Si può comunque utilizzare la macro `./compile Miao.kit` dalla directory `kitten_XXX`

Il mio primo programma Kitten

```
class Miao {  
    constructor() {}  
  
    method void main() {  
        "miao\n".output()  
    }  
}
```

- **Deve** essere contenuto nel file `Miao.kit`
- Una classe eseguibile deve definire il `main` e il costruttore senza parametri
- Se `Miao.kit` sta dentro `testcases` (consigliato) allora dovete entrare in `testcases` per compilarlo:

```
cd testcases; java -cp .. Kitten Miao.kit
```
- Si può comunque utilizzare la macro `./compile Miao.kit` dalla directory `kitten_XXX`

Il mio primo programma Kitten

```
class Miao {  
    constructor() {}  
  
    method void main() {  
        "miao\n".output()  
    }  
}
```

- **Deve** essere contenuto nel file `Miao.kit`
- Una classe eseguibile deve definire il `main` e il costruttore senza parametri
- Se `Miao.kit` sta dentro `testcases` (consigliato) allora dovete entrare in `testcases` per compilarlo:

```
cd testcases; java -cp .. Kitten Miao.kit
```
- Si può comunque utilizzare la macro `./compile Miao.kit` dalla directory `kitten_XXX`

Il mio secondo programma Kitten

```
class Led {  
    field boolean state  
  
    constructor() {}  
  
    method void on()  
        state := true  
  
    method void off()  
        state := false  
  
    method boolean isOn()  
        return state  
  
    method boolean isOff()  
        return !state  
}
```

- I metodi possono essere ereditati e ridefiniti
- I metodi sono sempre d'istanza (hanno un `this`)
- I costruttori sono invocati implicitamente dall'istruzione `new`
- I costruttori chiamano sempre implicitamente il costruttore della superclasse a zero argomenti

Metodi e costruttori

- I metodi possono essere ereditati e ridefiniti
- I metodi sono sempre d'istanza (hanno un `this`)
- I costruttori sono invocati implicitamente dall'istruzione `new`
- I costruttori chiamano sempre implicitamente il costruttore della superclasse a zero argomenti

Metodi e costruttori

- I metodi possono essere ereditati e ridefiniti
- I metodi sono sempre d'istanza (hanno un `this`)
- I costruttori sono invocati implicitamente dall'istruzione `new`
- I costruttori chiamano sempre implicitamente il costruttore della superclasse a zero argomenti

Metodi e costruttori

- I metodi possono essere ereditati e ridefiniti
- I metodi sono sempre d'istanza (hanno un `this`)
- I costruttori sono invocati implicitamente dall'istruzione `new`
- I costruttori chiamano sempre implicitamente il costruttore della superclasse a zero argomenti

Chiamate virtuali

```
class S {  
    constructor () {}  
  
    method String toString()  
        return "sono una S"  
}
```

```
class A extends S {  
    constructor () {}  
  
    method String  
        toString()  
        return "sono una A"  
}
```

```
class B extends S {  
    constructor () {}  
  
    method String  
        toString()  
        return "sono una B"  
}
```

Chiamate virtuali

```
class S {  
    constructor () {}  
  
    method String toString()  
        return "sono una S"  
}
```

```
class A extends S {  
    constructor () {}  
  
    method String  
        toString()  
        return "sono una A"  
}
```

```
class B extends S {  
    constructor () {}  
  
    method String  
        toString()  
        return "sono una B"  
}
```

Chiamate virtuali

```
class S {  
    constructor () {}  
  
    method String toString()  
        return "sono una S"  
}
```

```
class A extends S {  
    constructor () {}  
  
    method String  
        toString()  
        return "sono una A"  
}
```

```
class B extends S {  
    constructor () {}  
  
    method String  
        toString()  
        return "sono una B"  
}
```

Chiamate virtuali (cont.)

```
class List {
  field Object head
  field List tail
  constructor() {}
  constructor(Object head, List tail) {
    this.head := head;
    this.tail := tail }

  method void nomi() {
    head.toString().concat("\n").output();
    if (!(tail = nil)) then tail.nomi() }

  method void main() {
    List l := new List(new A(),
      new List(new B(),new List(new S(),nil)));
    l.nomi() }
}
```

Array e iterazione

```
class Arrays {
  field array of array of int aa
  field int MAX := 10
  constructor() {
    int i; int j;
    array of array of S bb :=
      [[new A()],[new B()]];

    aa := new array of int [MAX];
    for (i := 0; i < MAX; i := i + 1) {
      aa[i] := new int [i];
      for (j := 0; j < i; j := j + 1)
        aa[i][j] := i + j
    }
  }
}
```

La classe predefinita String

```
class Fibonacci {
    constructor() {}
    method int fib(int n)
        if (n = 0 | n = 1) then return 1
        else return this.fib(n - 1)
            + this.fib(n - 2)
    method void main() {
        String s := new String();
        "Immetti un numero: ".output();
        s.input();
        "Fibonacci(" .concat(s).concat(") = "
            .concat(this.fib(s.toInt()))).output();
        "\n".output()
    }
}
```

I metodi della classe String sono enumerati dentro
testcases/String.kit

La classe predefinita String

```
class Fibonacci {
    constructor() {}
    method int fib(int n)
        if (n = 0 | n = 1) then return 1
        else return this.fib(n - 1)
            + this.fib(n - 2)
    method void main() {
        String s := new String();
        "Immetti un numero: ".output();
        s.input();
        "Fibonacci(" .concat(s) .concat(") = "
            .concat(this.fib(s.toInt())) .output();
        "\n".output()
    }
}
```

I metodi della classe String sono enumerati dentro
testcases/String.kit

Tipici errori in Kitten

- La segnalazione di errore di sintassi di Kitten è un po' rudimentale
- Si ricordi che il `;` è un separatore di comandi
 - Quindi non separa campi né metodi
 - Non va messo alla fine di un metodo
 - Non va messo alla fine di un `then` se segue un `else`
 - Va messo dopo un condizionale o un ciclo se segue un altro comando
 - Separa dichiarazioni di variabili locali!
- Il condizionale richiede di usare la parola chiave `then`
- I campi vanno dichiarati con `field`, i metodi con `method` e i costruttori con `constructor`
- Il riferimento nullo è `nil`

Tipici errori in Kitten

- La segnalazione di errore di sintassi di Kitten è un po' rudimentale
- Si ricordi che il `;` è un separatore di comandi
 - Quindi non separa campi né metodi
 - Non va messo alla fine di un metodo
 - Non va messo alla fine di un `then` se segue un `else`
 - Va messo dopo un condizionale o un ciclo se segue un altro comando
 - Separa dichiarazioni di variabili locali!
- Il condizionale richiede di usare la parola chiave `then`
- I campi vanno dichiarati con `field`, i metodi con `method` e i costruttori con `constructor`
- Il riferimento nullo è `nil`

Tipici errori in Kitten

- La segnalazione di errore di sintassi di Kitten è un po' rudimentale
- Si ricordi che il `;` è un separatore di comandi
 - Quindi non separa campi né metodi
 - Non va messo alla fine di un metodo
 - Non va messo alla fine di un `then` se segue un `else`
 - Va messo dopo un condizionale o un ciclo se segue un altro comando
 - Separa dichiarazioni di variabili locali!
 - Il condizionale richiede di usare la parola chiave `then`
 - I campi vanno dichiarati con `field`, i metodi con `method` e i costruttori con `constructor`
 - Il riferimento nullo è `nil`

Tipici errori in Kitten

- La segnalazione di errore di sintassi di Kitten è un po' rudimentale
- Si ricordi che il `;` è un separatore di comandi
 - Quindi non separa campi né metodi
 - Non va messo alla fine di un metodo
 - Non va messo alla fine di un `then` se segue un `else`
 - Va messo dopo un condizionale o un ciclo se segue un altro comando
 - Separa dichiarazioni di variabili locali!
- Il condizionale richiede di usare la parola chiave `then`
- I campi vanno dichiarati con `field`, i metodi con `method` e i costruttori con `constructor`
- Il riferimento nullo è `nil`

Tipici errori in Kitten

- La segnalazione di errore di sintassi di Kitten è un po' rudimentale
- Si ricordi che il `;` è un separatore di comandi
 - Quindi non separa campi né metodi
 - Non va messo alla fine di un metodo
 - Non va messo alla fine di un `then` se segue un `else`
 - Va messo dopo un condizionale o un ciclo se segue un altro comando
 - Separa dichiarazioni di variabili locali!
- Il condizionale richiede di usare la parola chiave `then`
- I campi vanno dichiarati con `field`, i metodi con `method` e i costruttori con `constructor`
- Il riferimento nullo è `nil`

Tipici errori in Kitten

- La segnalazione di errore di sintassi di Kitten è un po' rudimentale
- Si ricordi che il `;` è un separatore di comandi
 - Quindi non separa campi né metodi
 - Non va messo alla fine di un metodo
 - Non va messo alla fine di un `then` se segue un `else`
 - Va messo dopo un condizionale o un ciclo se segue un altro comando
 - Separa dichiarazioni di variabili locali!
- Il condizionale richiede di usare la parola chiave `then`
- I campi vanno dichiarati con `field`, i metodi con `method` e i costruttori con `constructor`
- Il riferimento nullo è `nil`

Tipici errori in Kitten

- La segnalazione di errore di sintassi di Kitten è un po' rudimentale
- Si ricordi che il `;` è un separatore di comandi
 - Quindi non separa campi né metodi
 - Non va messo alla fine di un metodo
 - Non va messo alla fine di un `then` se segue un `else`
 - Va messo dopo un condizionale o un ciclo se segue un altro comando
 - Separa dichiarazioni di variabili locali!
- Il condizionale richiede di usare la parola chiave `then`
- I campi vanno dichiarati con `field`, i metodi con `method` e i costruttori con `constructor`
- Il riferimento nullo è `nil`

Tipici errori in Kitten

- La segnalazione di errore di sintassi di Kitten è un po' rudimentale
- Si ricordi che il `;` è un separatore di comandi
 - Quindi non separa campi né metodi
 - Non va messo alla fine di un metodo
 - Non va messo alla fine di un `then` se segue un `else`
 - Va messo dopo un condizionale o un ciclo se segue un altro comando
 - Separa dichiarazioni di variabili locali!
- Il condizionale richiede di usare la parola chiave `then`
- I campi vanno dichiarati con `field`, i metodi con `method` e i costruttori con `constructor`
- Il riferimento nullo è `nil`

Tipici errori in Kitten

- La segnalazione di errore di sintassi di Kitten è un po' rudimentale
- Si ricordi che il `;` è un separatore di comandi
 - Quindi non separa campi né metodi
 - Non va messo alla fine di un metodo
 - Non va messo alla fine di un `then` se segue un `else`
 - Va messo dopo un condizionale o un ciclo se segue un altro comando
 - Separa dichiarazioni di variabili locali!
- Il condizionale richiede di usare la parola chiave `then`
- I campi vanno dichiarati con `field`, i metodi con `method` e i costruttori con `constructor`
- Il riferimento nullo è `nil`

Tipici errori in Kitten

- La segnalazione di errore di sintassi di Kitten è un po' rudimentale
- Si ricordi che il `;` è un separatore di comandi
 - Quindi non separa campi né metodi
 - Non va messo alla fine di un metodo
 - Non va messo alla fine di un `then` se segue un `else`
 - Va messo dopo un condizionale o un ciclo se segue un altro comando
 - Separa dichiarazioni di variabili locali!
- Il condizionale richiede di usare la parola chiave `then`
- I campi vanno dichiarati con `field`, i metodi con `method` e i costruttori con `constructor`
- Il riferimento nullo è `nil`

Una *semplice* ricorsione...

Si completi la seguente classe Kitten:

```
class Hanoi {  
    constructor() {}  
  
    method void risolvi  
        (int h, int da, int a, int parcheggio)  
        ...  
  
    method void main()  
        this.risolvi(5,1,2,3)  
}
```

Quicksort

Si scriva una classe `Qsort.kit` con un metodo `sort` con la seguente intestazione:

```
method array of int  
sort(array of int a, int length)
```

Mergesort

Si scriva una classe `Msort.kit` con un metodo `sort` con la seguente intestazione:

```
method array of int  
sort(array of int a, int length)
```

Quicksort

Si scriva una classe `Qsort.kit` con un metodo `sort` con la seguente intestazione:

```
method array of int  
sort(array of int a, int length)
```

Mergesort

Si scriva una classe `Msort.kit` con un metodo `sort` con la seguente intestazione:

```
method array of int  
sort(array of int a, int length)
```