Class Cell

java.lang.Object
  Cell

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

Cell represents a single tile of a maze. STUDENT WILL NEED TO COMPLETE THIS CLASS AS INDICATED BELOW

Constructor Summary

Constructors

Constructor	Description
`Cell()`	Construct a cell that has all four walls by default, and which is given a String value that matches the uniquely- assigned number of the Cell.
`Cell(boolean n, boolean s, boolean e, boolean w)`	Cell constructor with explicit wall parameters, and which is given a String value that matches the uniquely- assigned number of the Cell.

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method	Description
`boolean`	`equals(Cell other)`	Check if two cells are the same in a grid, based on number only.
`boolean`	`equals(java.lang.Object other)`	Check if an Object is same as this cell, based on number only.
`java.lang.String`	`getData()`	Return the String data stored in this cell.
`int`	`getNumber()`	Get the cell number in the grid.
boolean	`hasEast()`	Return whether this cell's east wall exists.
boolean	`hasNorth()`	Return whether this cell's north wall exists.
boolean	`hasSouth()`	Return whether this cell's south wall exists.
boolean	`hasWest()`	Return whether this cell's west wall exists.
`java.lang.String`	`setData(java.lang.String contents)`	Set the String contents of this cell.
void	`setWalls(java.lang.String walls)`	Use a "bit" string in NESW (north-east-south-west) order to represent and set the walls of this cell.
`java.lang.String`	`toString()`	Get a "bit string" representation of this cell's walls, in NESW (north-east-south-west) order.

Methods inherited from class java.lang.Object

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

Constructor Details

Cell

```
public Cell()
```

Construct a cell that has all four walls by default, and which is given a String value that matches the uniquely- assigned number of the Cell.

Cell

```
public Cell(boolean n,
            boolean s,
            boolean e,
            boolean w)
```

Cell constructor with explicit wall parameters, and which is given a String value that matches the uniquely- assigned number of the Cell.

Parameters:

`n` - true if north side of the cell should have a wall

`s` - true if south side of the cell should have a wall

`e` - true if west side of the cell should have a wall

`e` - true if east side of the cell should have a wall

Method Details

setWalls

```
public void setWalls(java.lang.String walls)
```

Use a "bit" string in NESW (north-east-south-west) order to represent and set the walls of this cell. A 1 bit indicates that the wall exists, a 0 (or anything else) means no wall. The given string is assumed to have length at least 4; any characters in it beyond the first four will be ignored.

Parameters:

`walls` - the bit string to parse

toString

```
public java.lang.String toString()
```

Get a "bit string" representation of this cell's walls, in NESW (north-east-south-west) order. A 1 represents that a wall exists, and a 0 represents no wall. For example, "1001" is returned when only the north and west walls exist for a cell.

Overrides:

`toString` in class `java.lang.Object`

Returns:

the 4-character "bit string"

hasNorth

```
public boolean hasNorth()
```

Return whether this cell's north wall exists.

Returns:

true if and only if the north wall exists

hasEast

```
public boolean hasEast()
```

Return whether this cell's east wall exists.

Returns:

true if and only if the east wall exists

hasSouth

```
public boolean hasSouth()
```

Return whether this cell's south wall exists.

Returns:

true if and only if the south wall exists

hasWest

```
public boolean hasWest()
```

Return whether this cell's west wall exists.

Returns:

true if and only if the west wall exists

getData

```
public java.lang.String getData()
```

Return the String data stored in this cell.

Returns:

the data value

setData

```
public java.lang.String setData(java.lang.String contents)
```

Set the String contents of this cell.

Parameters:

`contents` - the cell's new data

Returns:

the original contents

getNumber

```
public int getNumber()
```

Get the cell number in the grid.

Returns:

the number

equals

```
public boolean equals(Cell other)
```

Check if two cells are the same in a grid, based on number only.

Parameters:

`other` - the other cell to compare to this

Returns:

true if same, false otherwise

equals

```
public boolean equals(java.lang.Object other)
```

Check if an Object is same as this cell, based on number only. If the Object is not a cell, then return false. THIS METHOD IS COMPLETE; STUDENT SHOULD NOT MODIFY IT.

Overrides:

`equals` in class `java.lang.Object`

Parameters:

`other` - the other cell to compare to this

Returns:

true if same, false otherwise