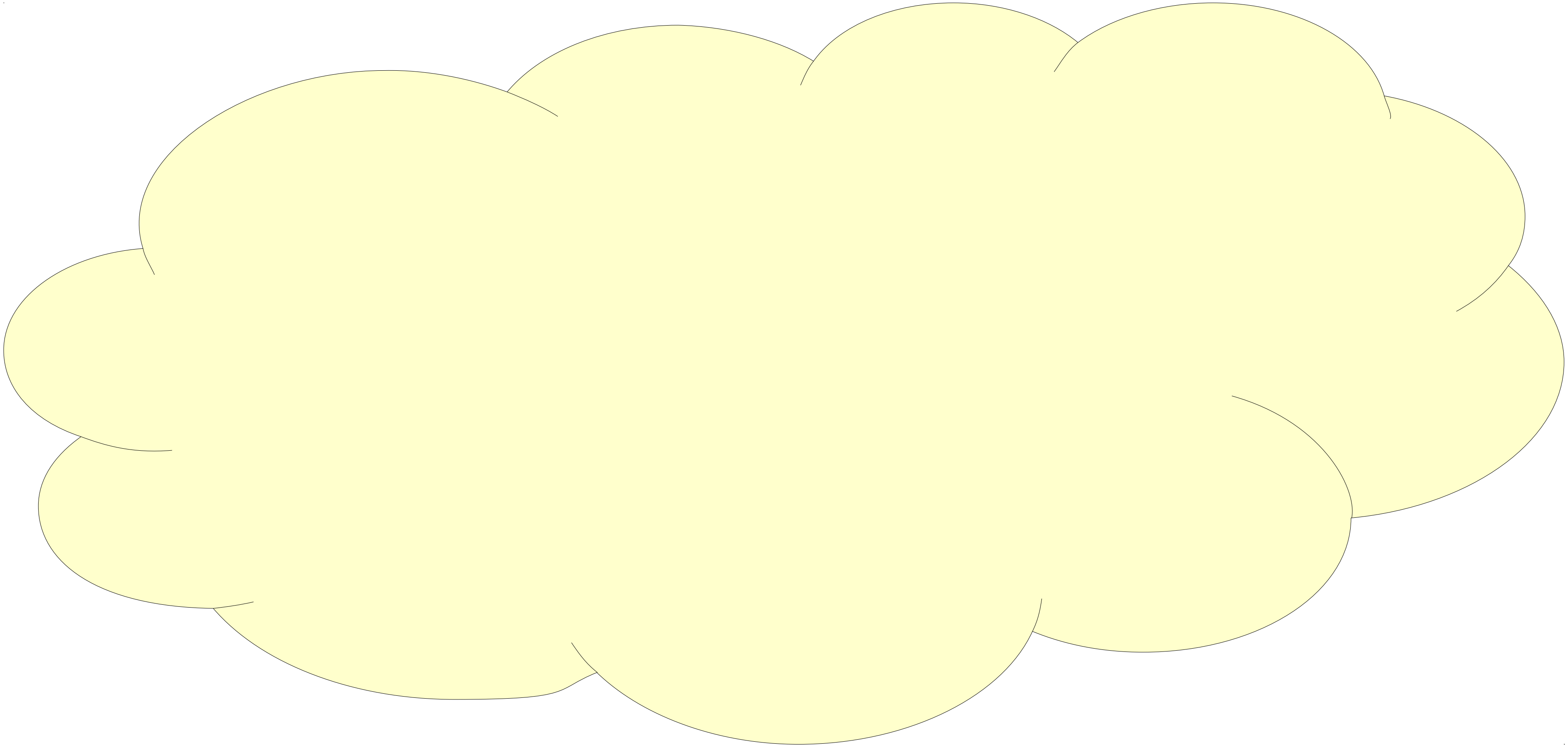
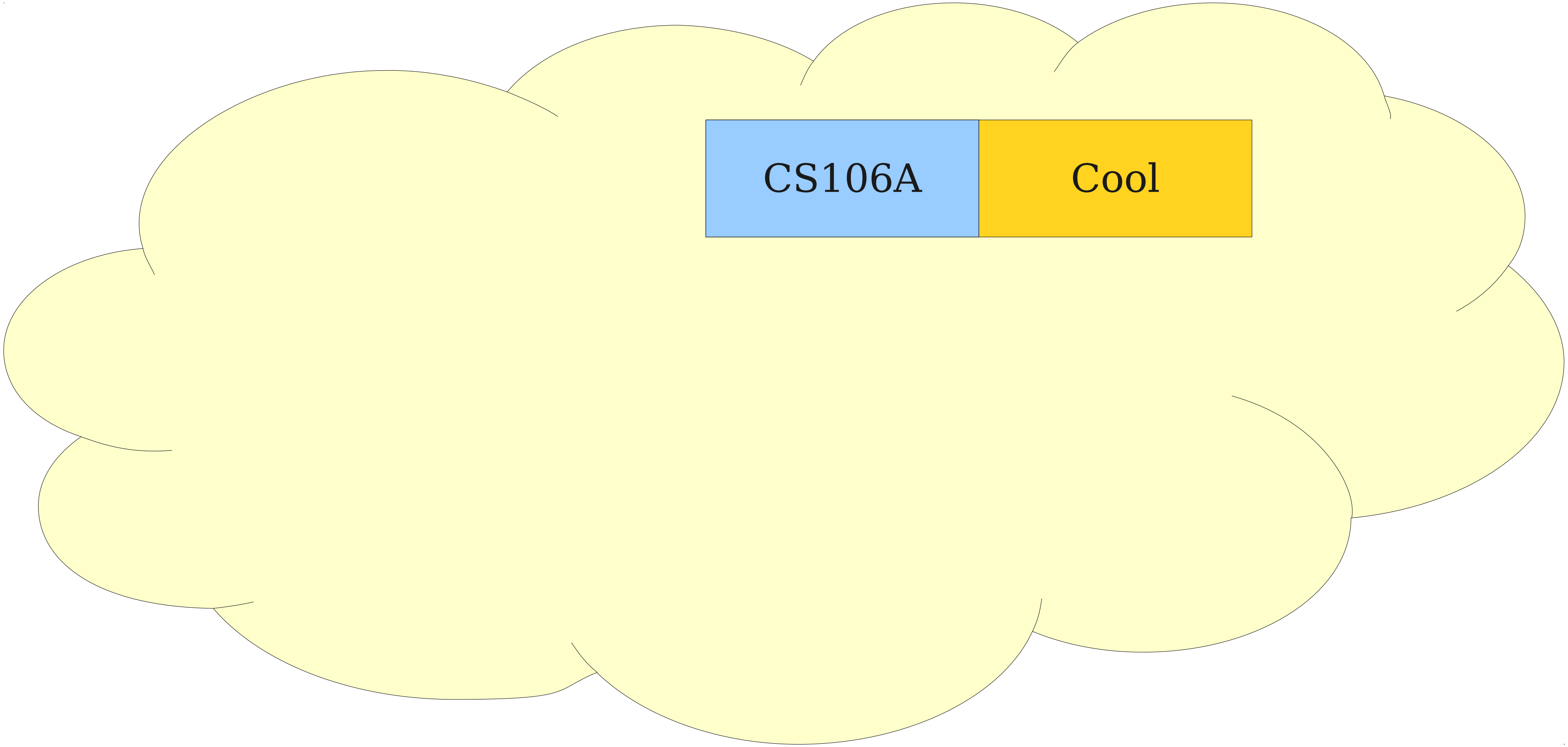
HashMap

Friday Four Square Today!
Outside Gates at 4:15PM

Not All Data is Linear



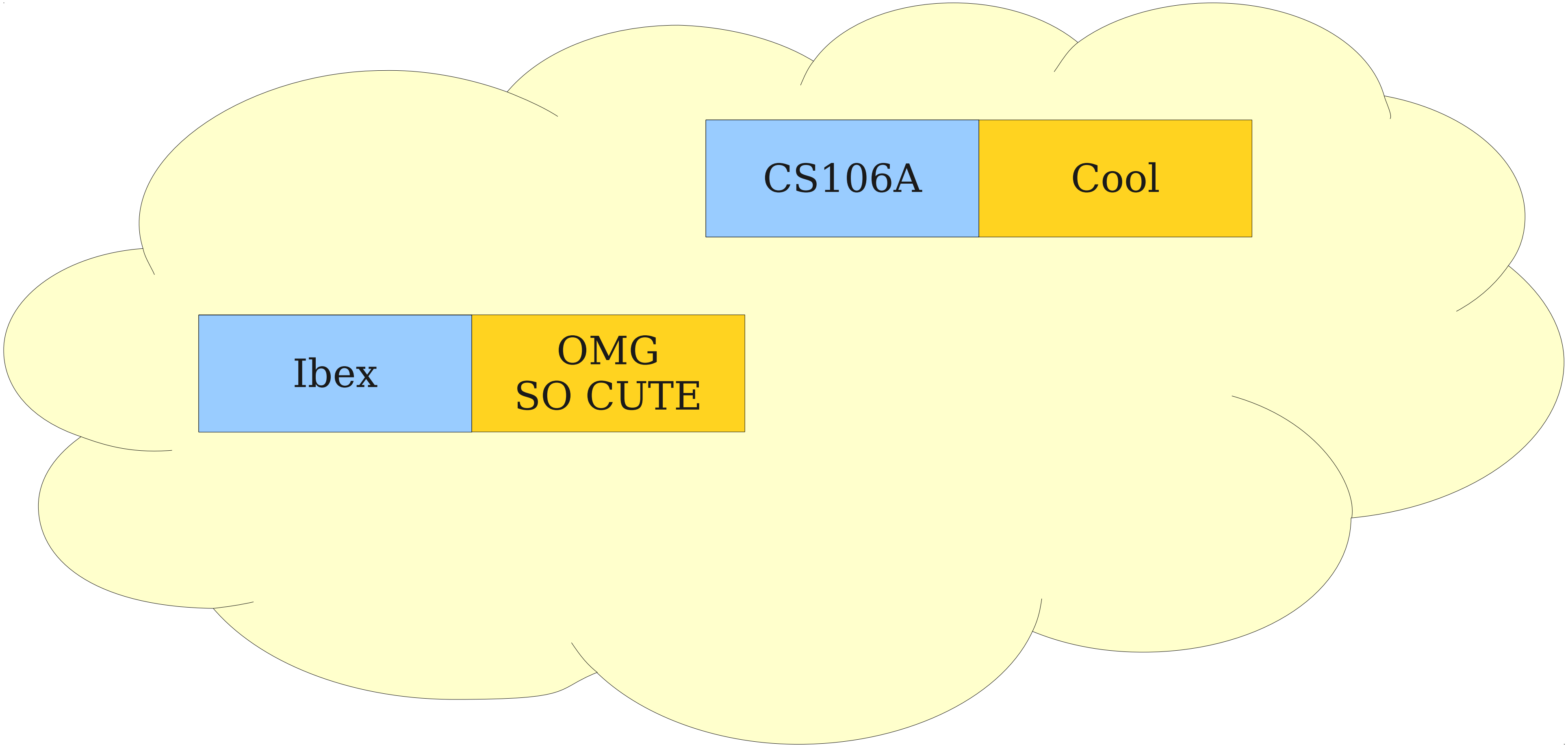
```
HashMap<String, String> myMap = new HashMap<String, String>();
```



```
HashMap<String, String> myMap = new HashMap<String, String>();
myMap.put("CS106A", "Cool");
```

To add a key/value pair to a **HashMap**, use the syntax

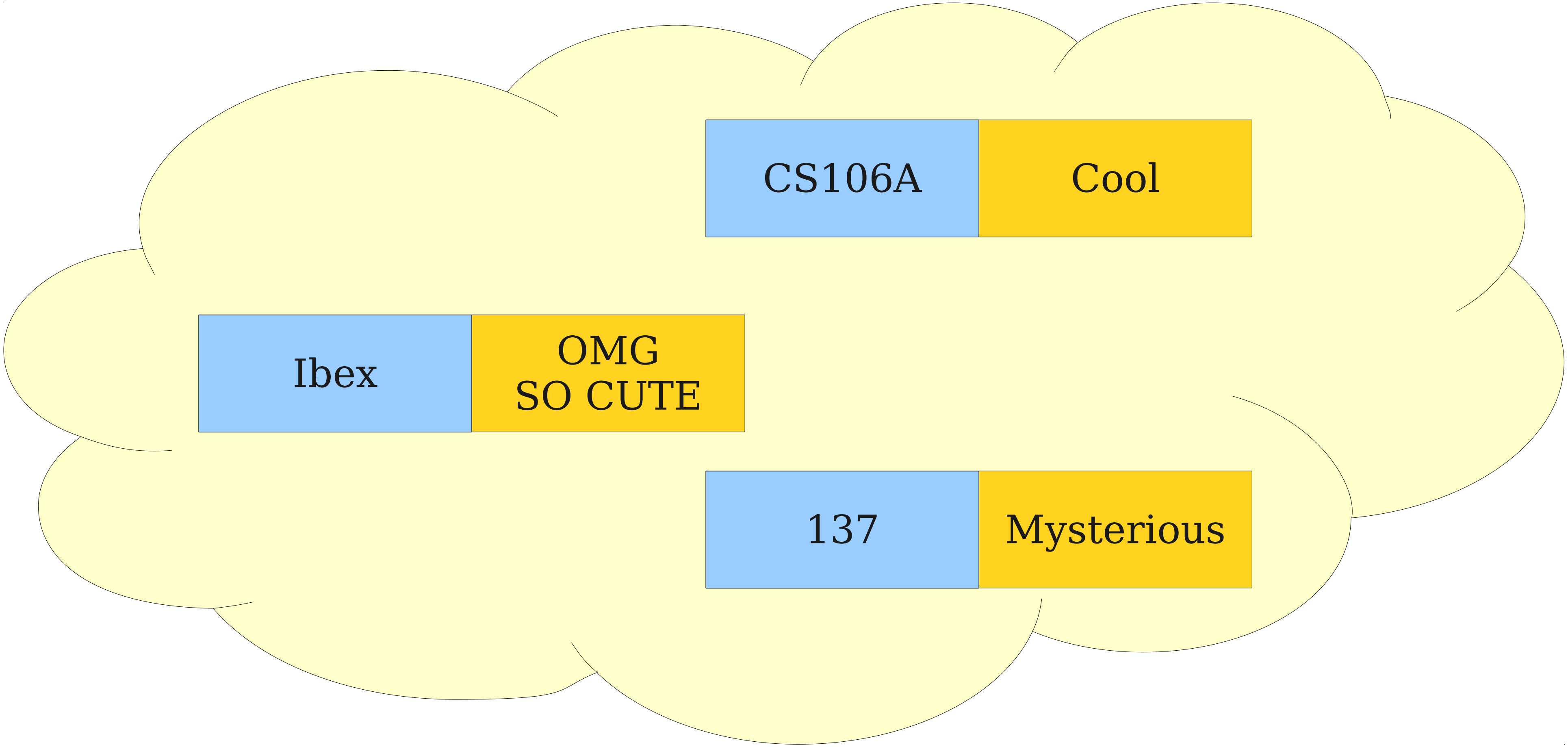
***map*.put(*key*, *value*)**



CS106A Cool

Ibex OMG SO CUTE

```
HashMap<String, String> myMap = new HashMap<String, String>();
myMap.put("CS106A", "Cool");
myMap.put("Ibex", "OMG SO CUTE");
```



CS106A

Cool

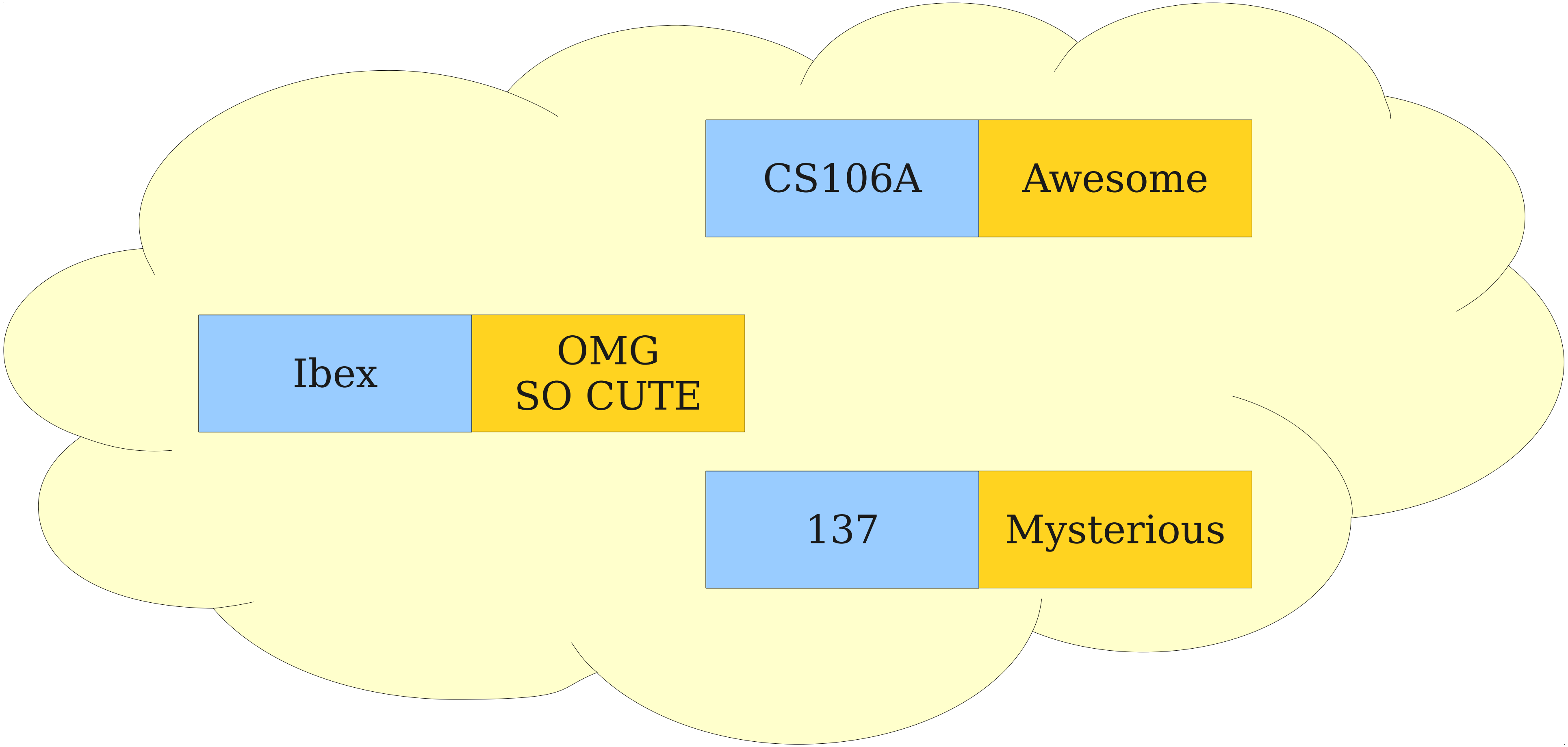
Ibex

OMG
SO CUTE

137

Mysterious

```
HashMap<String, String> myMap = new HashMap<String, String>();
myMap.put("CS106A", "Cool");
myMap.put("Ibex", "OMG SO CUTE");
myMap.put("137", "Mysterious");
```



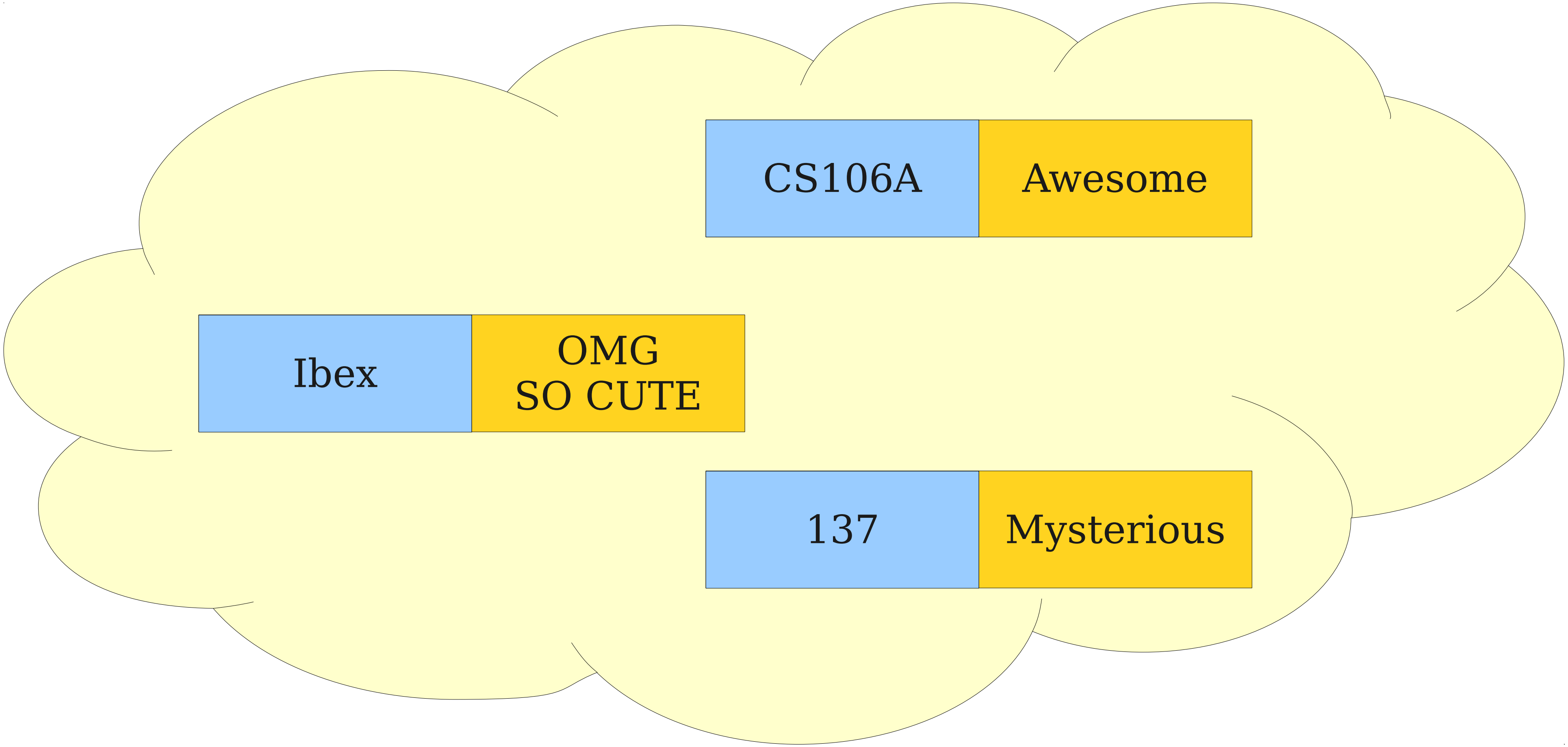
CS106A Awesome

Ibex OMG SO CUTE

137 Mysterious

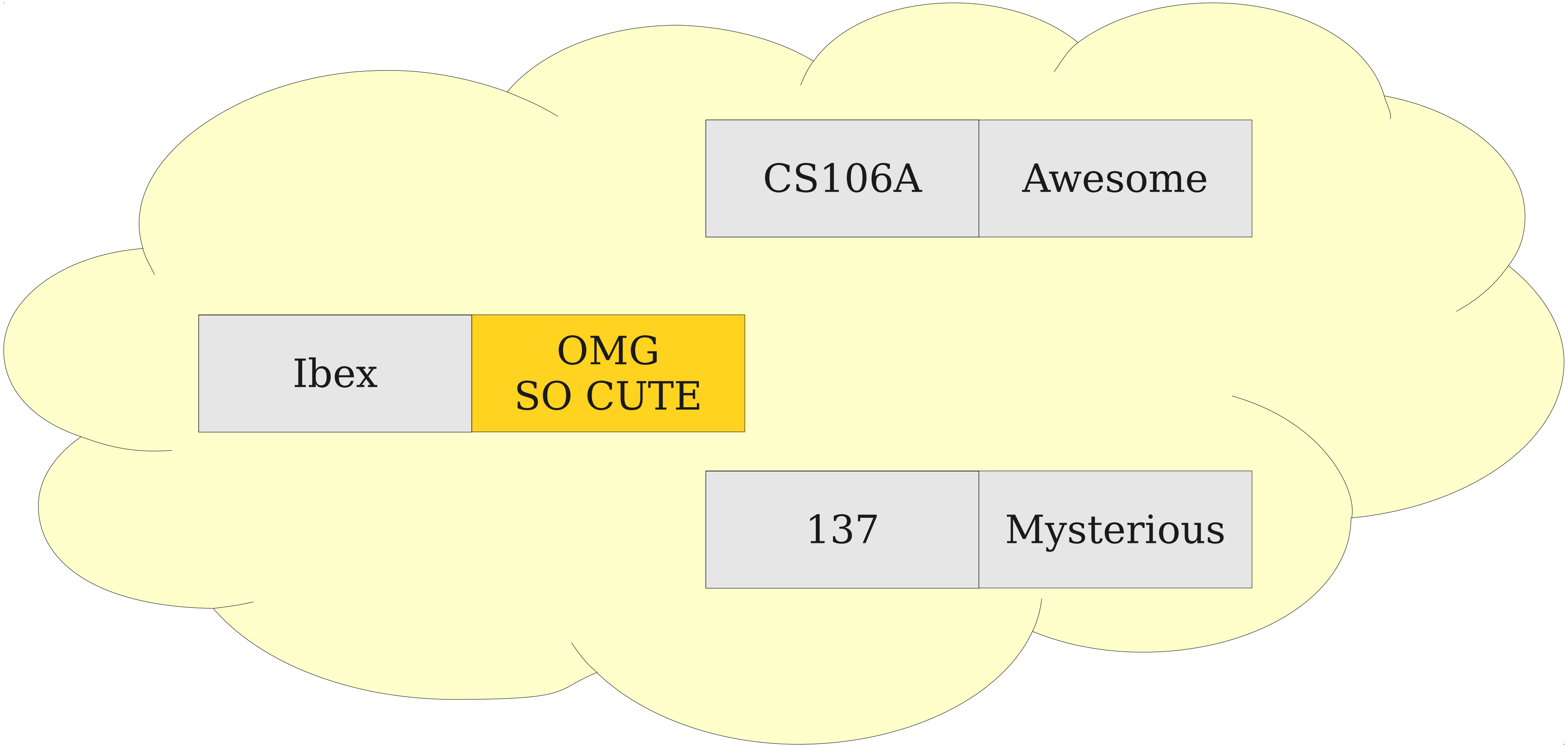
```
HashMap<String, String> myMap = new HashMap<String, String>();
myMap.put("CS106A", "Cool");
myMap.put("Ibex", "OMG SO CUTE");
myMap.put("137", "Mysterious");
myMap.put("CS106A", "Awesome");
```

If you **put** a key/value pair where the key exists, the old value is replaced.



```
HashMap<String, String> myMap = new HashMap<String, String>();
myMap.put("CS106A", "Cool");
myMap.put("Ibex", "OMG SO CUTE");
myMap.put("137", "Mysterious");
myMap.put("CS106A", "Awesome");

myMap.get("Ibex");
```

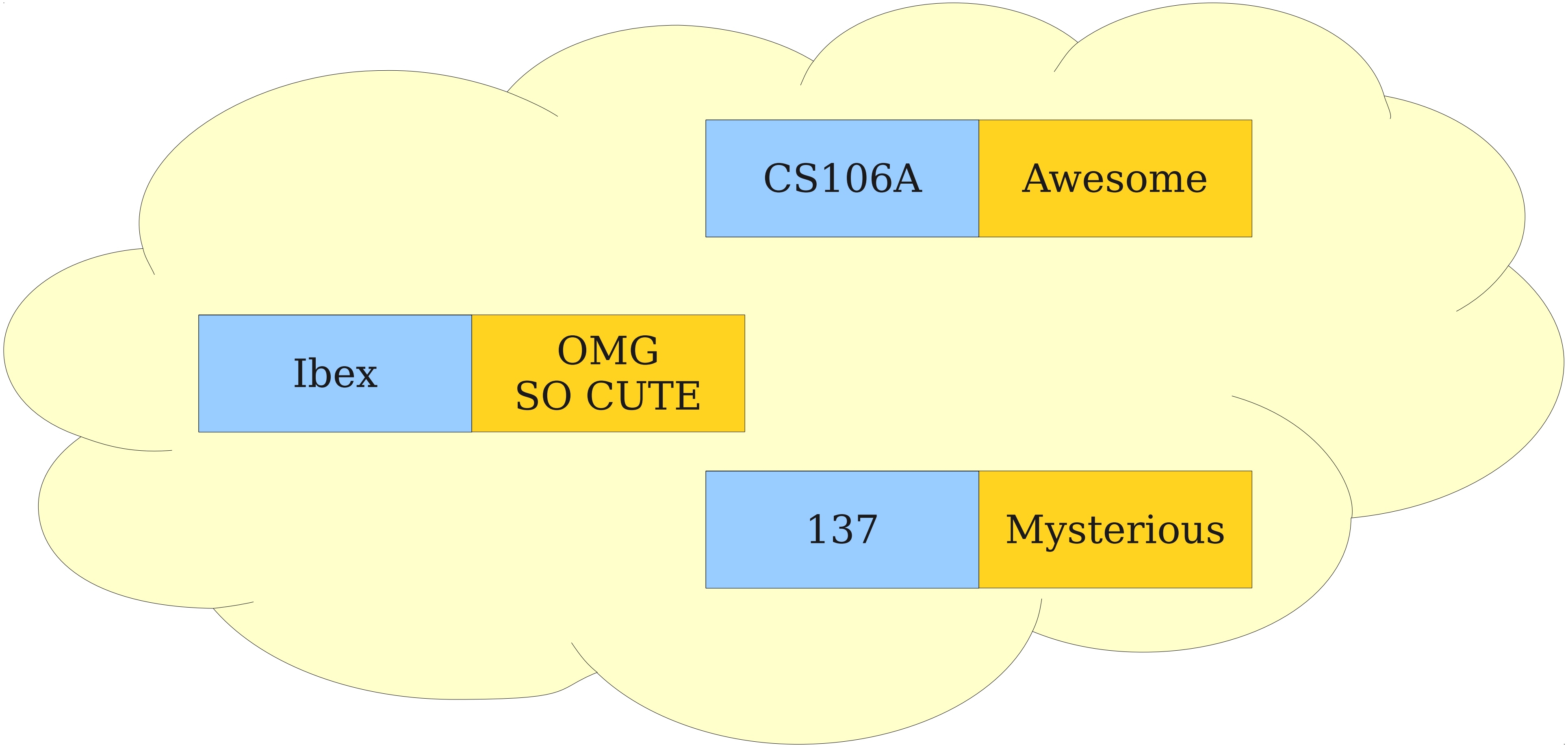


```
HashMap<String, String> myMap = new HashMap<String, String>();
myMap.put("CS106A", "Cool");
myMap.put("Ibex", "OMG SO CUTE");
myMap.put("137", "Mysterious");
myMap.put("CS106A", "Awesome");

myMap.get("Ibex");
```

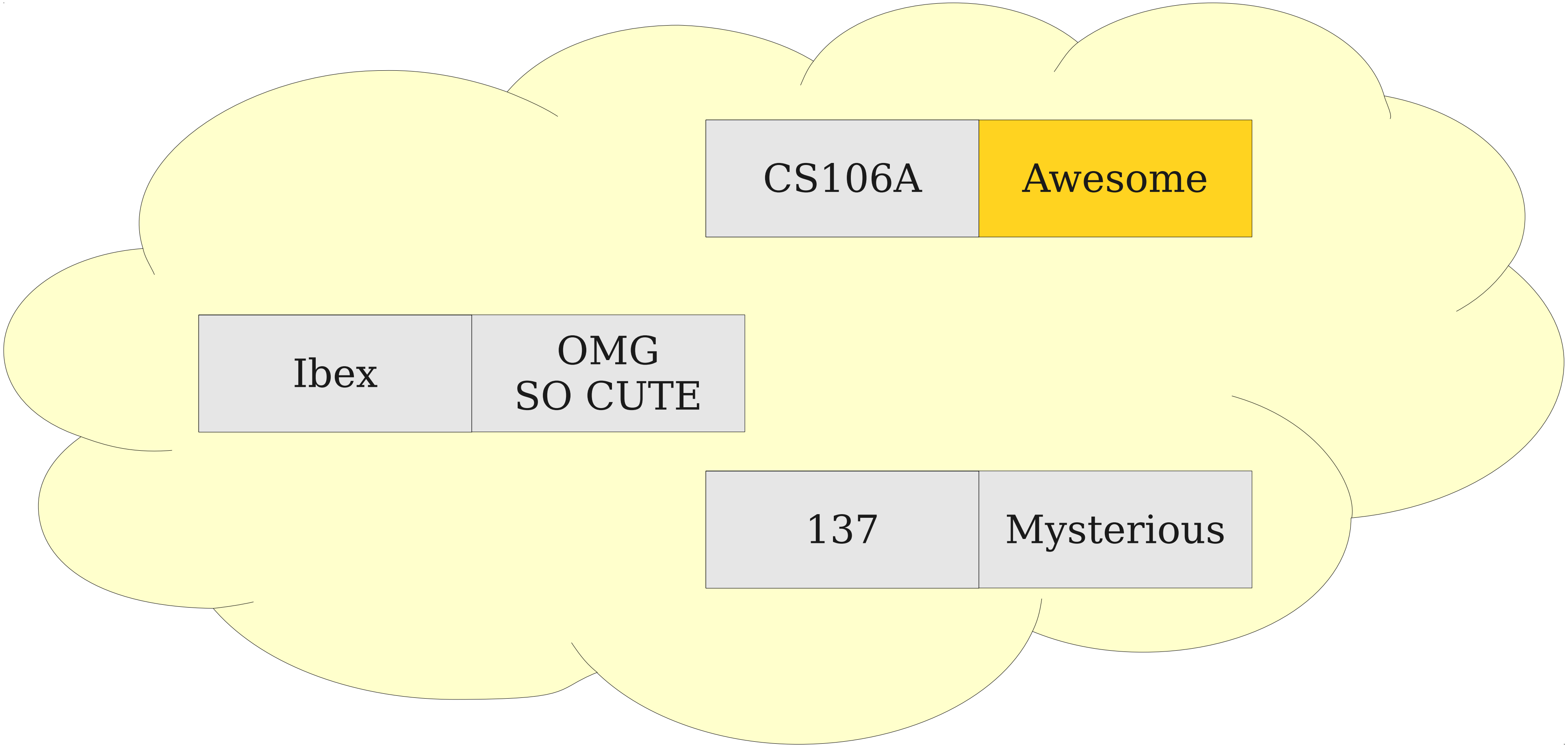
To look up the value associated with a key:

***map*.get(*key*)**



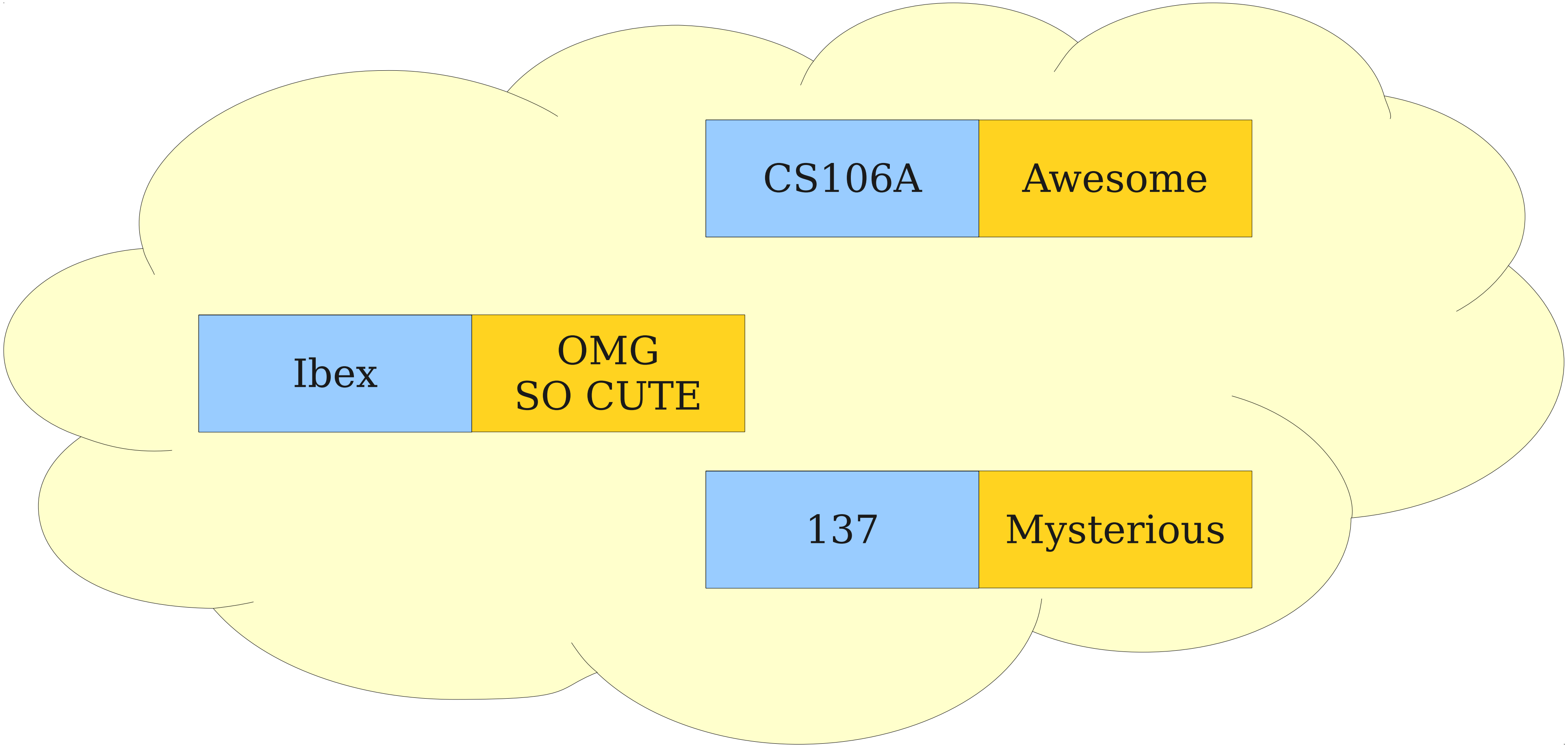
```
HashMap<String, String> myMap = new HashMap<String, String>();
myMap.put("CS106A", "Cool");
myMap.put("Ibex", "OMG SO CUTE");
myMap.put("137", "Mysterious");
myMap.put("CS106A", "Awesome");

myMap.get("Ibex");
myMap.get("CS106A");
```



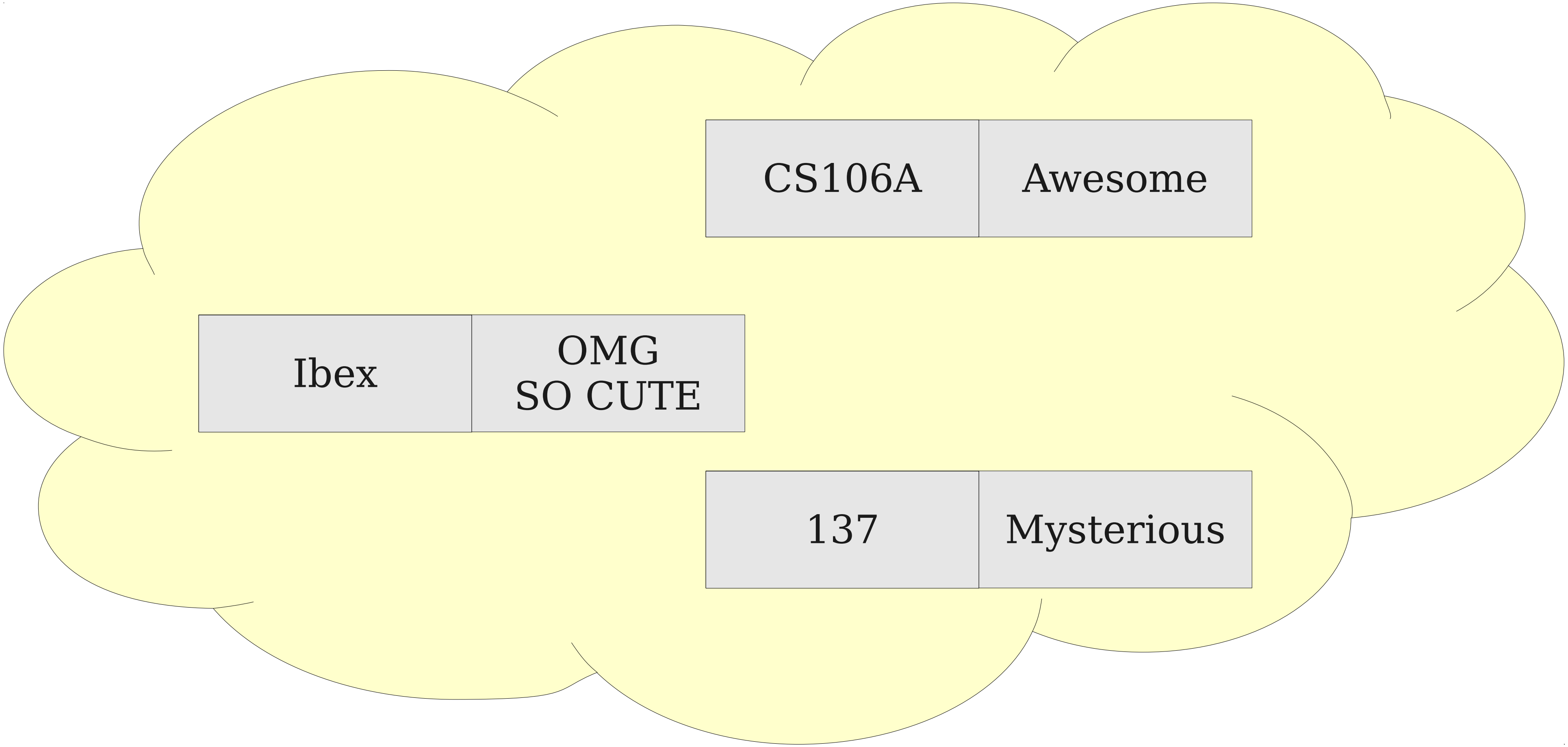
```
HashMap<String, String> myMap = new HashMap<String, String>();
myMap.put("CS106A", "Cool");
myMap.put("Ibex", "OMG SO CUTE");
myMap.put("137", "Mysterious");
myMap.put("CS106A", "Awesome");

myMap.get("Ibex");
myMap.get("CS106A");
```



```
HashMap<String, String> myMap = new HashMap<String, String>();
myMap.put("CS106A", "Cool");
myMap.put("Ibex", "OMG SO CUTE");
myMap.put("137", "Mysterious");
myMap.put("CS106A", "Awesome");

myMap.get("Ibex");
myMap.get("CS106A");
myMap.get("KE$HA");
```

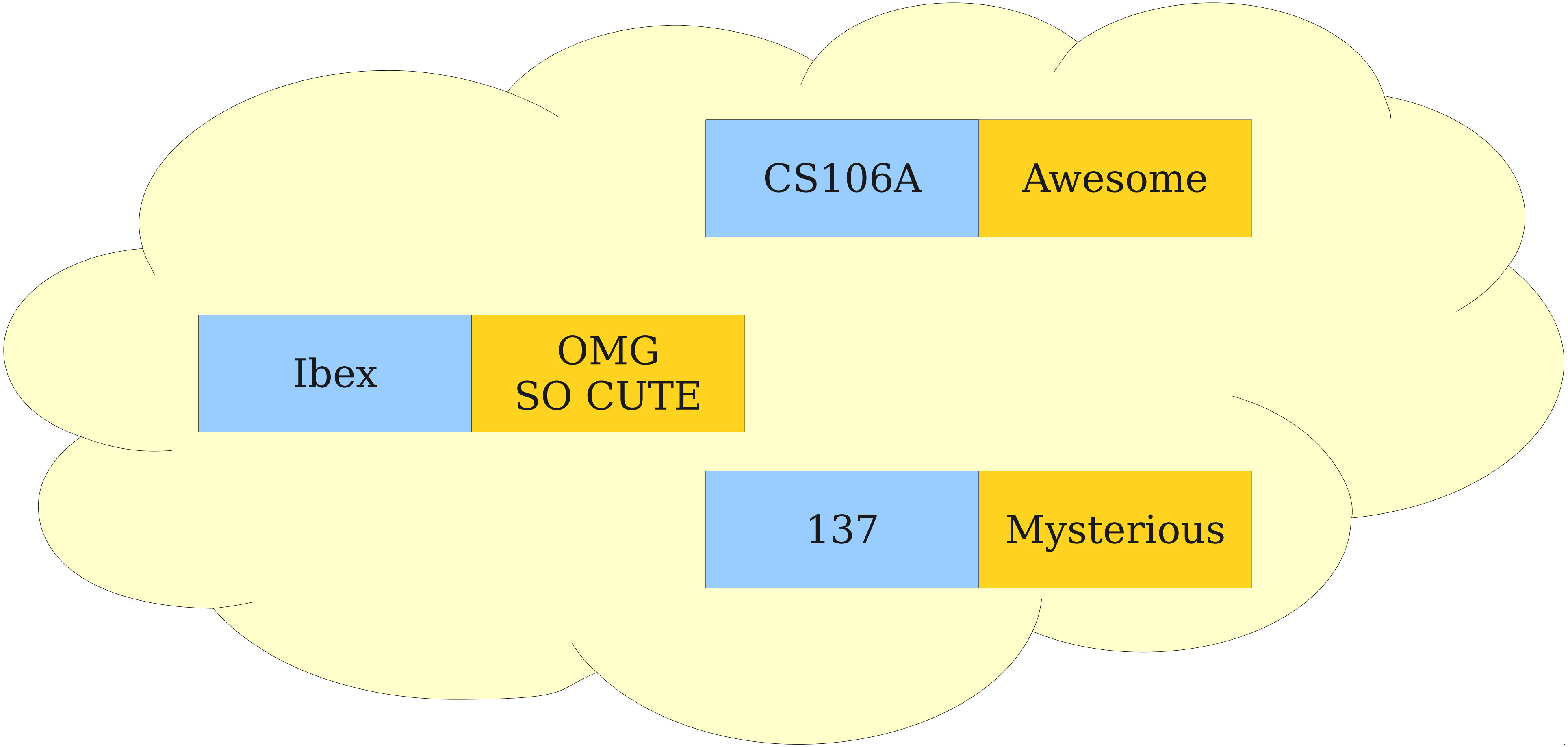


CS106A	Awesome
Ibex	OMG SO CUTE
137	Mysterious

```
HashMap<String, String> myMap = new HashMap<String, String>();
myMap.put("CS106A", "Cool");
myMap.put("Ibex", "OMG SO CUTE");
myMap.put("137", "Mysterious");
myMap.put("CS106A", "Awesome");

myMap.get("Ibex");
myMap.get("CS106A");
myMap.get("KE$HA");
```

If you **get** a key that isn't in a map, the method returns **null**.



```
HashMap<String, String> myMap = new HashMap<String, String>();
myMap.put("CS106A", "Cool");
myMap.put("Ibex", "OMG SO CUTE");
myMap.put("137", "Mysterious");
myMap.put("CS106A", "Awesome");

myMap.get("Ibex");
myMap.get("CS106A");
myMap.get("KE$HA");
myMap.containsKey("137");
```

You can check whether a key exists in the map:

***map*.containsKey(*key*)**

Basic `HashMap` Operations

- HashMap has two type arguments:

  `HashMap<`***KeyType***`,` ***ValueType***`>`

- To insert a key/value pair:

  map`.put(`***key***`,` ***value***`)`

- To look up the value associated with a key:

  map`.get(`***key***`)`

- To check whether a key exists:

  map`.containsKey(`***key***`)`

Making **`HashMap`** Shine

Exploring the US

Making Music

The Keyboard File Format

note-file-name
x
y
width
height
is white key?

The xkcd Color Survey

The xkcd Color Survey

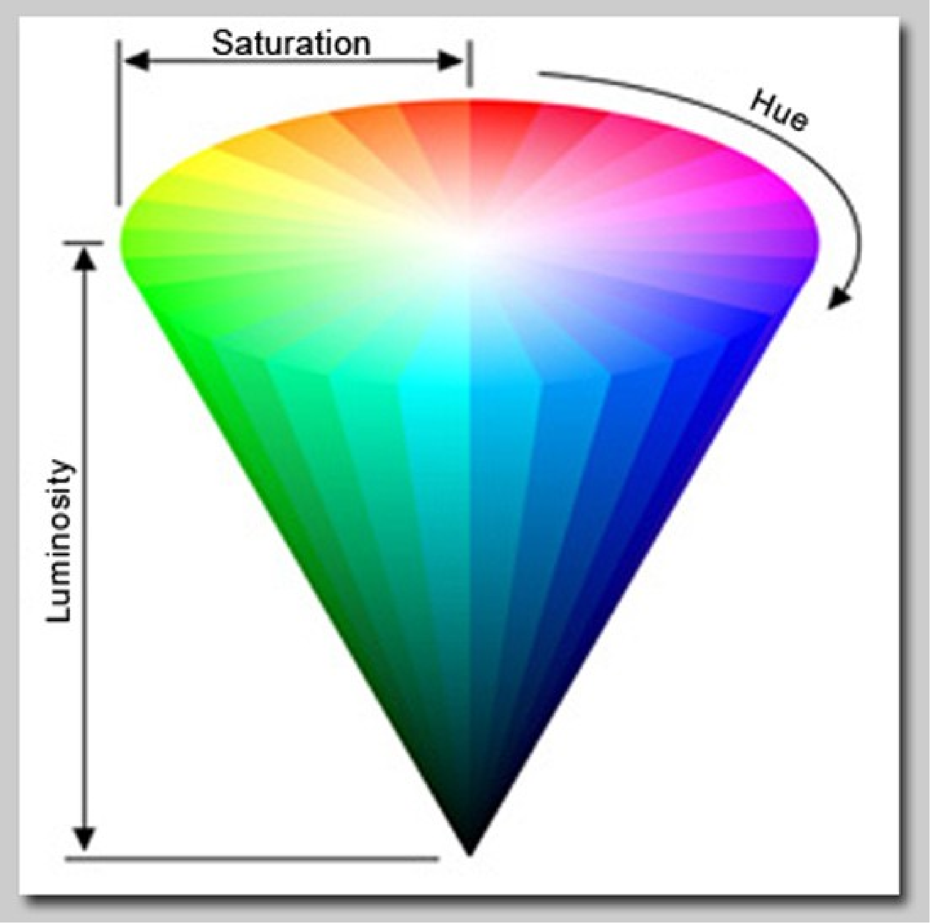
- Volunteers (online) were shown a randomly-chosen color and asked to name the color.
- The result is (after filtering) about 2.8 million RGB triplets and their names.
- What do people think the colors are?

The Color File Format

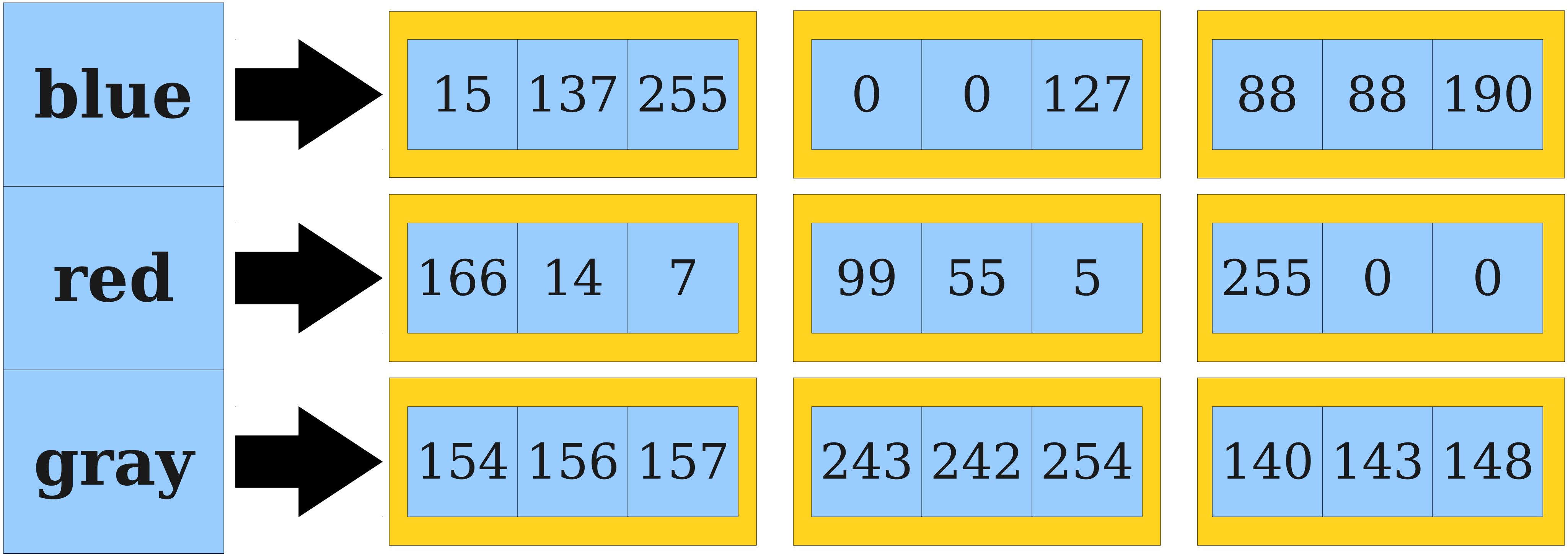
color-name
red
green
blue

Displaying Colors

- The HSB Color Format
  - Choose the **hue** (what color), **saturation** (how intense), and **brightness** (absolute brightness).
  - Each choice in the range from 0.0 to 1.0.



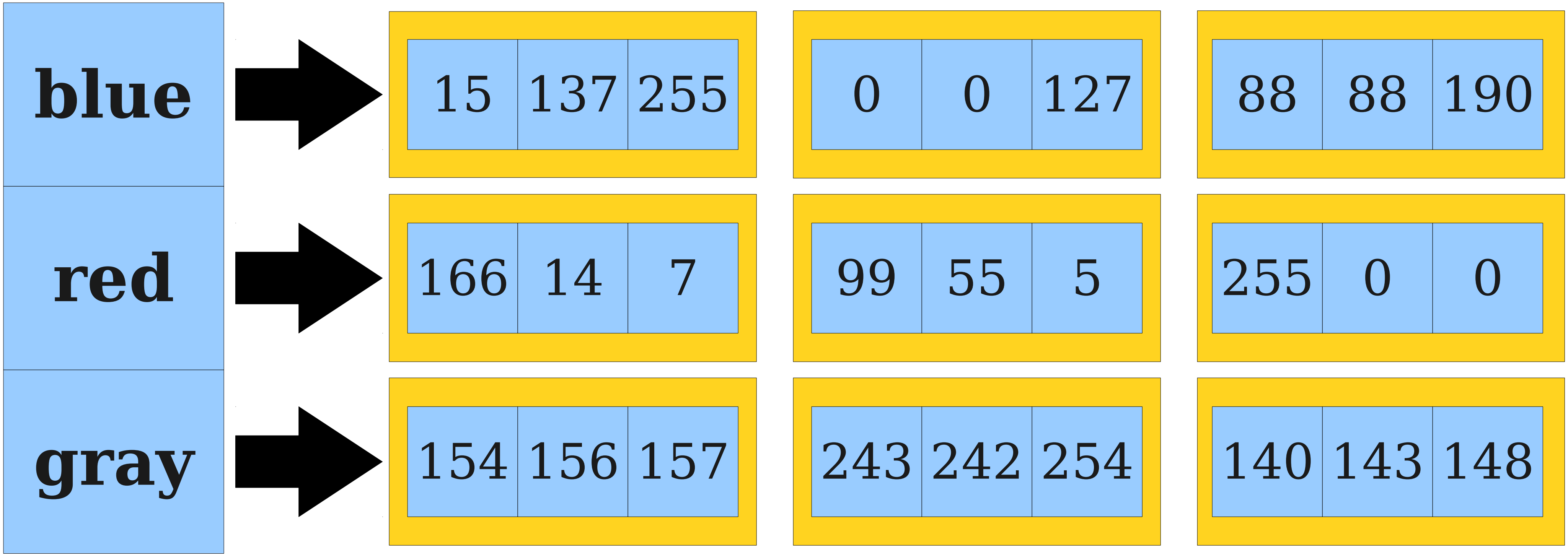
How to Structure the Data?



blue	15	137	255	0	0	127	88	88	190
red	166	14	7	99	55	5	255	0	0
gray	154	156	157	243	242	254	140	143	148

associate each color name with a list of RGB triplets

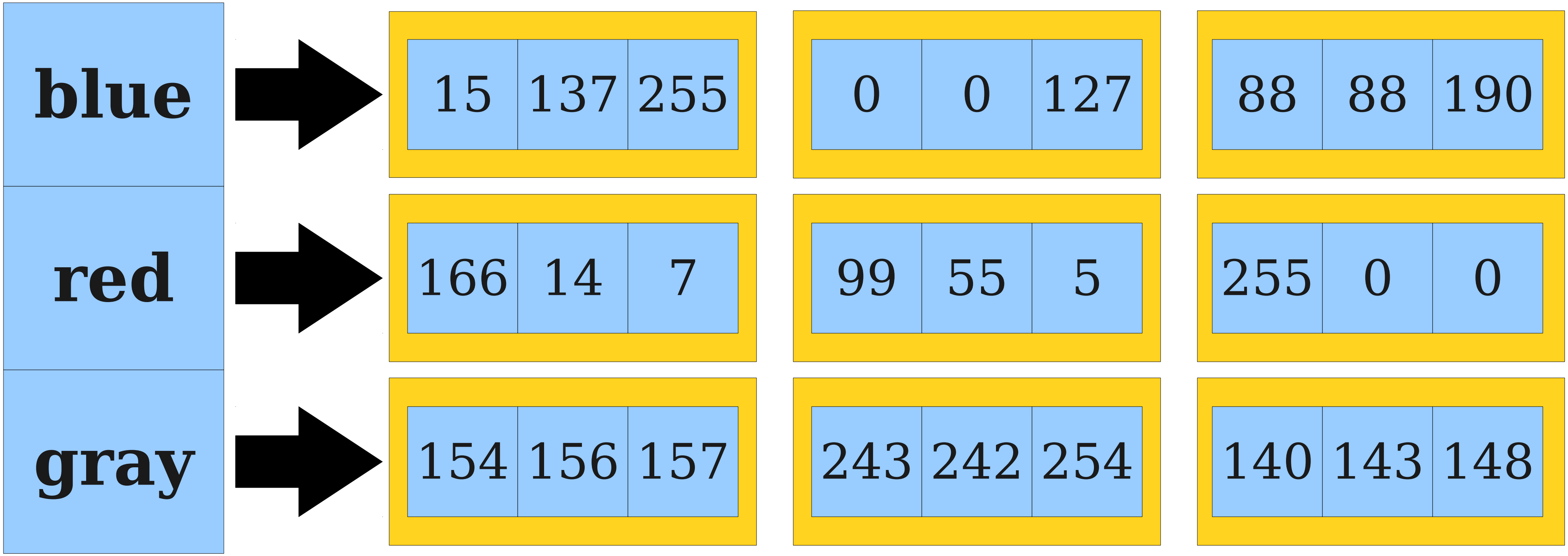
How to Structure the Data?



blue	15 137 255	0 0 127	88 88 190
red	166 14 7	99 55 5	255 0 0
gray	154 156 157	243 242 254	140 143 148

HashMap<*color name* , *list of RGB triplets*>

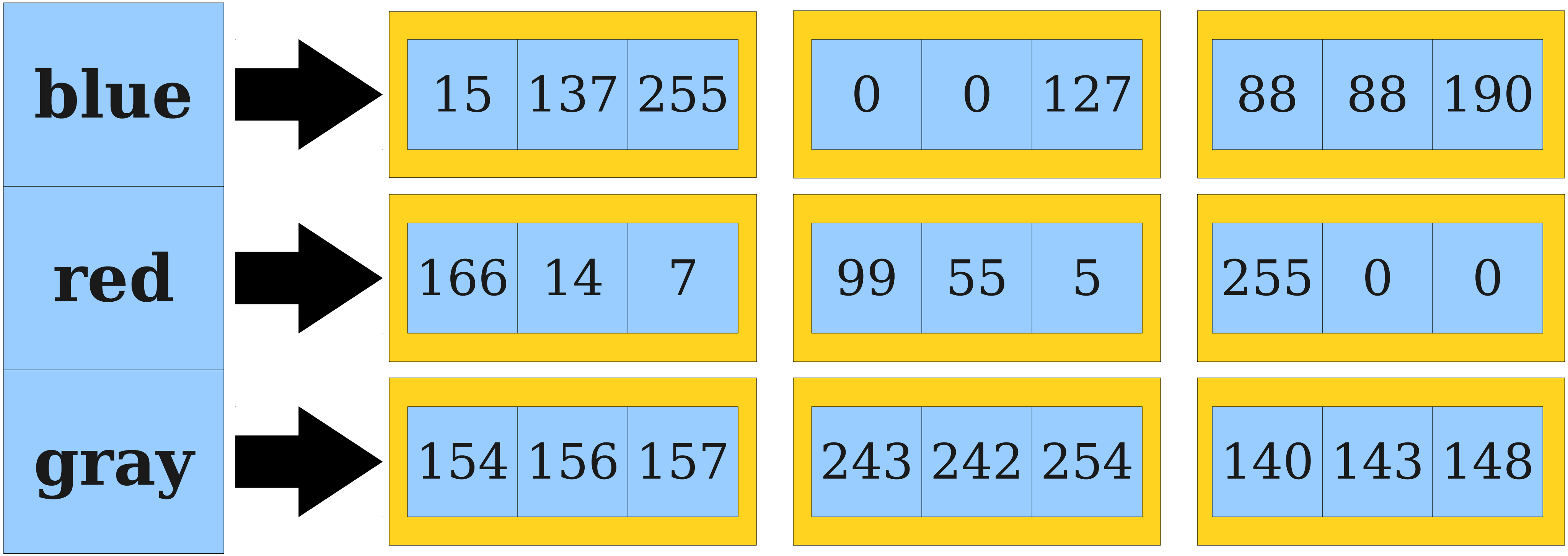
How to Structure the Data?



blue	15 137 255	0 0 127	88 88 190
red	166 14 7	99 55 5	255 0 0
gray	154 156 157	243 242 254	140 143 148

`HashMap<String,` ***list of RGB triplets*** **`>`**

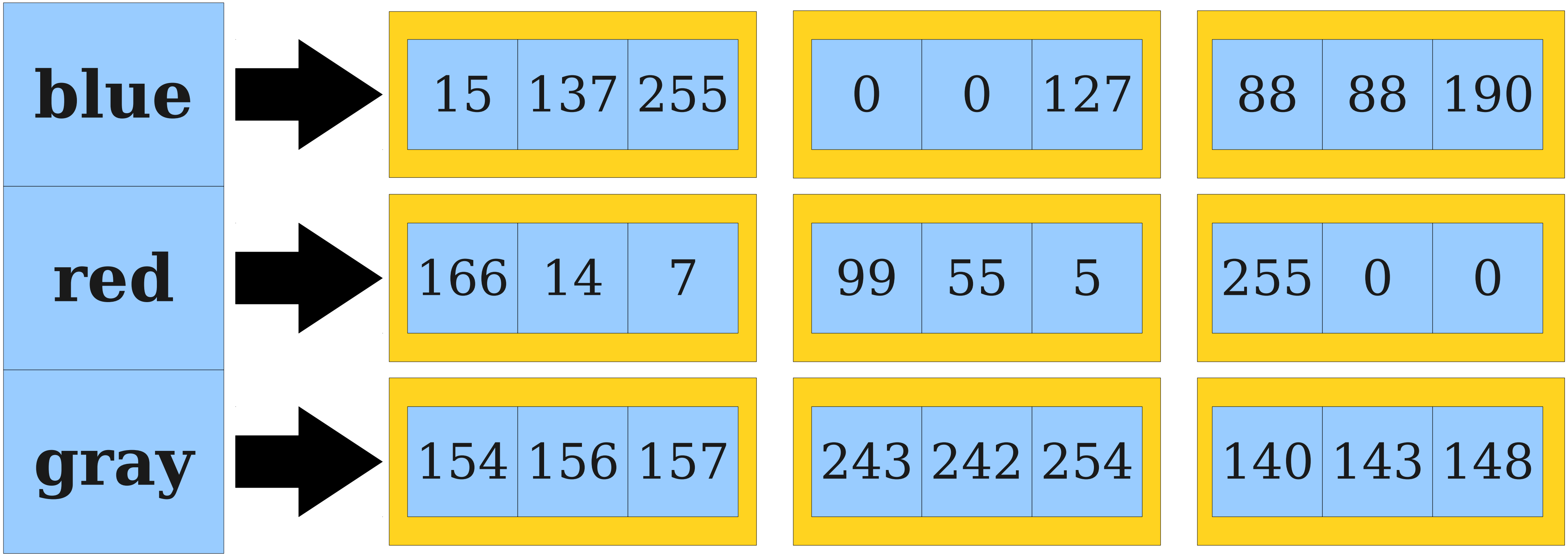
How to Structure the Data?



blue	15	137	255	0	0	127	88	88	190
red	166	14	7	99	55	5	255	0	0
gray	154	156	157	243	242	254	140	143	148

`HashMap<String, ArrayList<`***RGB triplet***`>>`

How to Structure the Data?



blue	15	137	255	0	0	127	88	88	190
red	166	14	7	99	55	5	255	0	0
gray	154	156	157	243	242	254	140	143	148

```
HashMap<String, ArrayList<int[]>>
```

For More Information

http://blog.xkcd.com/2010/05/03/color-survey-results/