

# UX/UI

## User Experience



# Today

- UX
  - Usability
  - Tutorial
  - Juiciness

```
* The coordinates (0, 0, 0) represents the octocube
*/
class GeoOctocube {
    /**
     * Gets the sector from the (x, y, z) specified
     *
     * Sector will be:
     * <code>
```

	5	6	
-	-	-	-
	7	8	2
	3	4	

```
* @param int $x the x coordinate
* @param int $y the y coordinate
* @param int $z the z coordinate
* @return int the number of the sector (0 if x =
```

```
static function get_sector($x, $y, $z) {
```

```
* Common fields
format = format[0]
regionnumbers = [] * placeholder
keytype = "" * placeholder
```

```
* Ducky heuristic to distinguish known_hosts from known_hosts2:
is second field entirely decimal digits?
if not (isdec($fields[1]))
```

```
* Treat all keys as host keys
* Format: hostkey: [keyid: [comment]]
* (PUTTY doesn't store the type of bits)
regionnumbers = map (long: $fields[1])
keytype = "ssh"
```

```
function getSector($x, $y, $z) {
    $x = floor($x / 5);
    $y = floor($y / 5);
    $z = floor($z / 5);
    $sector = ($x * 5 + $y * 5 + $z) / 5;
    return $sector;
}
```



# According to Norman & Nielsen...

“The first requirement for an exemplary user experience is to meet the exact needs of the customer, without fuss or bother.

Next comes simplicity and elegance that produce products that are a joy to own, a joy to use.”

<https://www.nngroup.com/articles/definition-user-experience/>

# How will the player receive your game?

- Usability
- Playability
- Stability



# How will the player receive your game?

- Usability
- Playability
- Stability



# How will the player receive your game?

- Usability
- Playability
- Stability



```
* @param int $x the x coordinate  
* @param int $y the y coordinate  
* @param int $z the z coordinate  
* @return int the number of the sector (0 if x =  
static function get_sector ($x, $y, $z) {
```



# Do they know what to do?

- What to click
- How to play the game
  - Tutorial

```
 * The coordinates (0, 0, 0) represents the octocube
 */
class GeoOctocube {
    /**
     * Gets the sector from the (x, y, z) specified
     *
     * Sector will be:
     * <code>
```

		5	6	
		-	-	
	7	8		2
	3	4		

```
 * @param int $x the x coordinate
 * @param int $y the y coordinate
 * @param int $z the z coordinate
 * @return int the number of the sector (0 if x =
static function get_sector($x, $y, $z) {
```



# What to click

```

    * Common fields
    Password = fields[0]
    regionnumbers = [] * placeholder
    keytype = "" * placeholder
  
```

```

    * Dorky heuristic to distinguish known_hosts from known_hosts2.
    // is second field entirely decimal digits?
    if (isdecimal(fields[1]))
    {
      * Travel all the way to host key
      * Format: hostkey, 255 exp10 mod10 comment
      * (PUTT doesn't store the size of bits)
      regionnumbers = new long [fields[1].length];
      region = "raw";
    }
  
```

```

    * Used by the
    * parameter $x the x coordinate
    * parameter $y the y coordinate
    * parameter $z the z coordinate
    * @return int the number of the sector (0 if x =
  
```

```

    * The coordinates (0, 0, 0) represents the octocube
    */
    class GeoOctocube {
  
```

```

    * Gets the sector from the (x, y, z) specified
  
```

```

    * Sector will be:
    * <code>
  
```

	5	6	
	-	-	-
7	8		2
3	4		

```

    * parameter $x the x coordinate
    * parameter $y the y coordinate
    * parameter $z the z coordinate
  
```

```

    * @return int the number of the sector (0 if x =
  
```

```

    function getSector($x, $y, $z)
    {
      $x = intval($x);
      $y = intval($y);
      $z = intval($z);
      // ...
    }
  
```





HINT

Wine Glasses  
Sledgehammer  
Old TV

Old Phone  
Old Flag  
Lighter

Flintlock Rifle  
Deco Table Lamp  
Collector Plate (3)

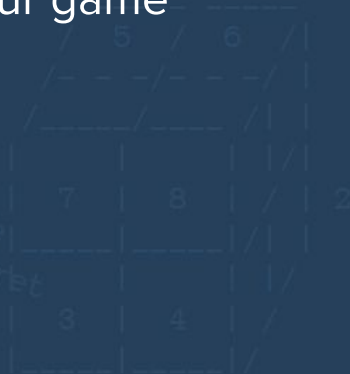
Chest  
Lantern  
Cigarette Case

ITEMS

MENU

# UIs must be clear

- Their purpose is to *help*
- Players should always know how to interact with your game
- Don't give them too much to focus on at once
- Create hierarchies, highlighting important things

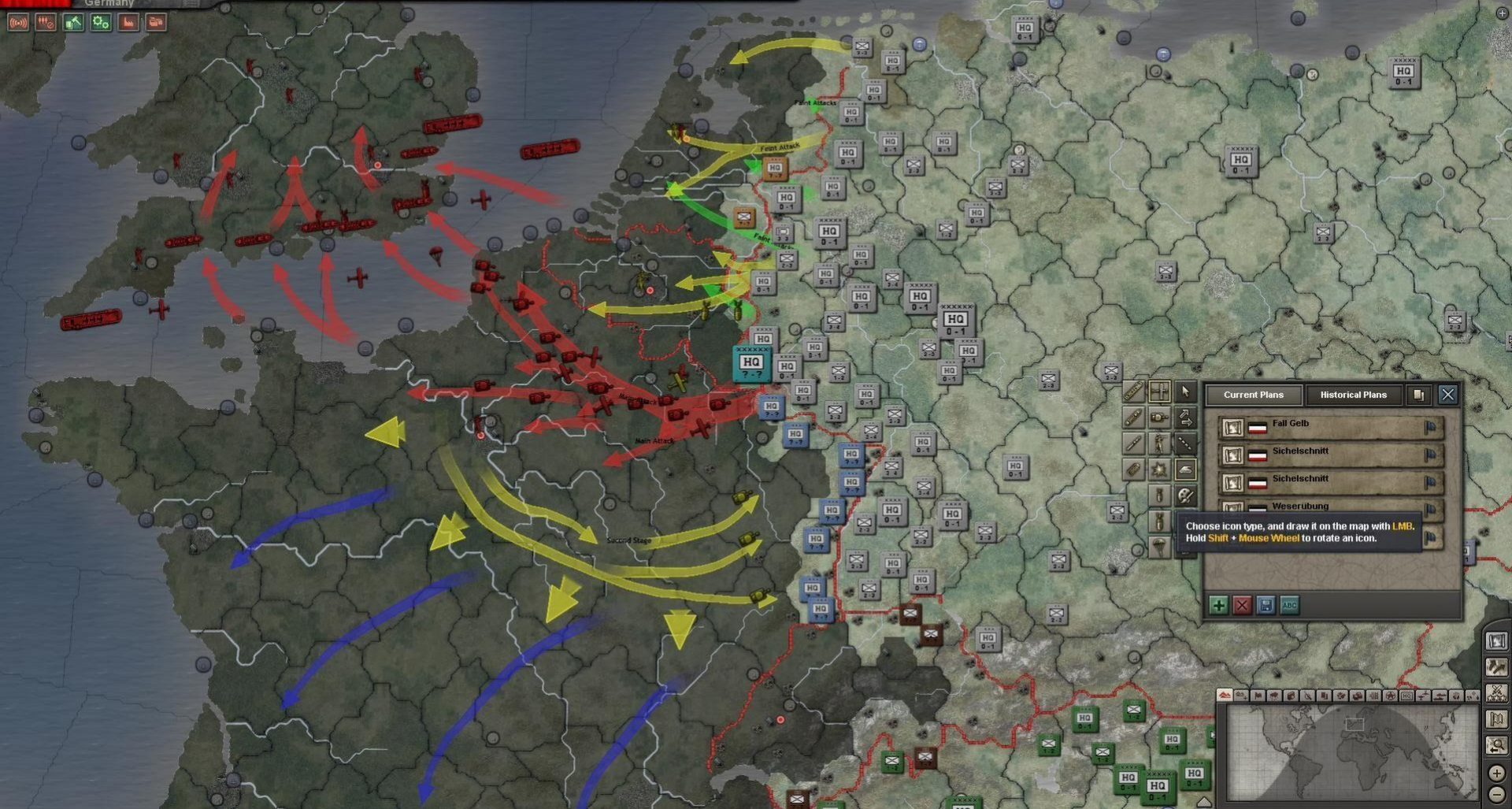


```

* The coordinates (0, 0, 0) represents the octocube
*/
class GeoOctocube {
    /**
     * Gets the sector from the (x, y, z) specified
     *
     * Sector will be:
     * <code>
     *
     * @param int $x the x coordinate
     * @param int $y the y coordinate
     * @param int $z the z coordinate
     * @return int the number of the sector (0 if x =
     */
    static function get_sector ($x, $y, $z) {
        // ...
    }
}

```





Current Plans Historical Plans

- Fall Gelb
- Sichelschnitt
- Sichelschnitt
- Weserübung

Choose icon type, and draw it on the map with LMB.  
Hold Shift + Mouse Wheel to rotate an icon.

+ X L ABC





100%

8

13K

8/0/10

29/32

0

Frankfurt

Prague

Luxemburg

Paris

Berne

Lyon

Marseille

Budapest

9:00, 6 Jan, 1936





Fleet Combat

**1st Husk Squadron**

**Ekir Poxinikis**  
Skill ★★★  
Retreat

- Corvette NKS Opalc D ...  
Corvette
- Corvette NKS Knak Spe ...  
Corvette
- Corvette NKS Knak Kra ...  
Corvette
- Corvette NKS Hulkrab M ...  
Corvette

0	0	596	100%
216	0	408	42%

**1st Stellar Cluster**

**Mashgirig**  
Skill ★

- Corvette MSS Hiraemen ...  
Corvette
- Corvette MSS Murgare ...  
Corvette
- Corvette MSS Uldoreem ...  
Corvette
- Corvette MSS Vabreem ...  
Corvette

0	216	0	57%
596	0	406	0%





```
require 'base32'
require 'cryptosphere'

module Cryptosphere
  # Blocks are the...
  # Common fields
  attr_accessor :x, :y, :z
  attr_accessor :name, :color, :material, :type, :value

  # Dirty heuristic to distinguish known hosts from known hosts2.
  # Second field entirely decimal digits?
  # (0-9)
  def self.valid?(x, y, z)
    # First field must not be 0
    # Second field must not be 0
  end
end
```

```
The coordinates (0, 0, 0) represents the octocube
*/
class GeoOctocube {
  # Gets the sector from the (x, y, z) specified
  #
  # @return int the number of the sector (0 if x =
  #
  # static function get_sector ($x, $y, $z) {
```



```
use when calculating the ID of a block
"crypt-block"
it on the size of...
1_048_576
urgence...
param int $x the x coordinate
param int $y the y coordinate
param int $z the z coordinate
@return int the number of the sector (0 if x =
static function get_sector ($x, $y, $z) {
```

# Draw attention

- Lights
- Particle effect
- Animations
- etc

```
* The coordinates (0, 0, 0) represents the octocube
*/
class GeoOctocube {
```

```
/**
 * Gets the sector from the (x, y, z) specified

```

```
 * Sector will be:
```

```
 * <code>
```

	5	6	
-	-	-	-
7	8		2
3	4		

```
 * @param int $x the x coordinate
 * @param int $y the y coordinate
 * @param int $z the z coordinate
```

```
 * @return int the number of the sector (0 if x =
```

```
static function get_sector ($x, $y, $z) {
```













Pay Respects

F

Press F to Pay Respects



# 8 bad ways of making a tutorial

1. Force the player to take the tutorial
2. Make the player read a lot
3. Describe buttons and menu items badly
4. Leave steps out
5. Punish inexperience
6. Patronize/humiliate the player
7. Force the player to complete the tutorial
8. Don't give them a tutorial at all



Also bad...

Have a tutorial that is completely separate from the rest of the game.

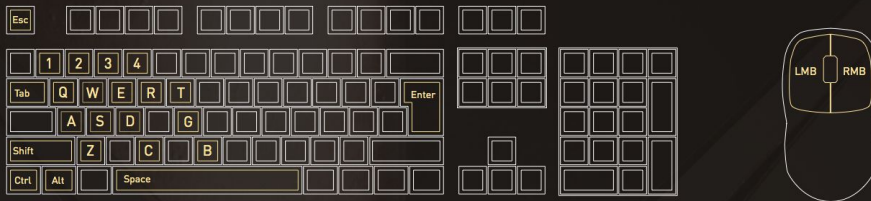


```
param int $x the x coordinate
param int $y the y coordinate
param int $z the z coordinate
@return int the number of the sector (0 if x =
static function get_sector ($x, $y, $z) {
```





# KEYBOARD AND MOUSE LAYOUT



## MOVEMENT

W	Move Forward
A	Strafe Left
S	Move Backward
D	Strafe Right
Spacebar	Jump
Shift	Travel Mode

## COMBAT

LMB	Basic Attack	Ctrl + Ability Key	Upgrade Ability
RMB	Ability 1	Left Ctrl	Cancel Ability
Q	Ability 2		
E	Ability 3		
R	Ability 4 (Ultimate)		

## ACTIVE CARDS

1	Card Slot 1
2	Card Slot 2
3	Card Slot 3
4	Card Slot 4

## GENERAL

B	Recall
C	Team Comm
G	Card Shop
Tab	Scoreboard
Alt + Z	Taunt
Enter	Team Chat
Tab	Cycle Channel





require 'base32'  
require 'cryptosphere' # primitives, blocks  
module Cryptosphere  
# Blocks are the underlying convergent  
# Cryptosphere for data confidentiality  
# sauce, welcome my friend, you have fo  
Blocks provide for both data integrity  
between 0 bytes and 1048576 bytes (1 me  
convergent encryption technique known  
approach, a cryptographic hash of the fi  
together with an optional random key know  
symmetric key is derived, which is used  
authenticated symmetric cipher.  
# more specifics on the encryption, please  
# key to use when calculating the ID of a block  
# matches the URI scheme for blocks  
= "crypt\_block"  
# ID on the size of a block  
= 1\_048\_576

```
# Common fields  
portal = Portal.new  
regionnumbers = [] # placeholders  
laytype = "" # placeholders
```

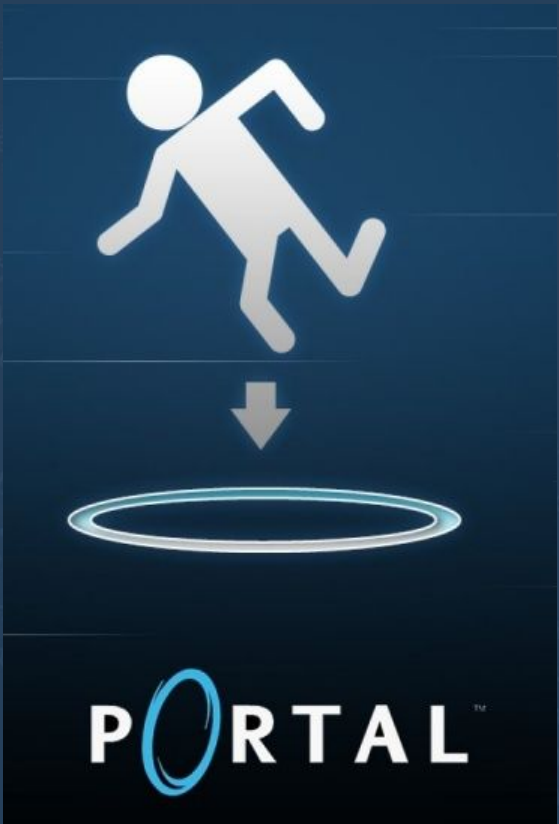
```
# Ducky heuristic to distinguish  
# a second field entirely, use  
# (r'^(?<=)fields$'  
# Treat all fields the same  
# Format: host:port  
# PUTTY down to state like  
regionnumbers = [] # long  
regionnumbers = ""
```

The coordinates (0, 0, 0) represents the octocube  
\*/  
class GeoOctocube {  
sector from the (x, y, z) specified  
ll be:  

	5	6	
	-	-	-
7	8		2
3	4		

```
function return sector from the  
geo octocube coordinates (x, y, z)  
[name, x, y, z] = Portal.new  
return sector from the (x, y, z)  
# 0 0 0  
# 0 0 1  
# 0 1 0  
# 0 1 1  
# 1 0 0  
# 1 0 1  
# 1 1 0  
# 1 1 1  
# 2 0 0  
# 2 0 1  
# 2 1 0  
# 2 1 1  
# 3 0 0  
# 3 0 1  
# 3 1 0  
# 3 1 1  
# 4 0 0  
# 4 0 1  
# 4 1 0  
# 4 1 1  
# 5 0 0  
# 5 0 1  
# 5 1 0  
# 5 1 1  
# 6 0 0  
# 6 0 1  
# 6 1 0  
# 6 1 1  
# 7 0 0  
# 7 0 1  
# 7 1 0  
# 7 1 1  
# 8 0 0  
# 8 0 1  
# 8 1 0  
# 8 1 1  
# 9 0 0  
# 9 0 1  
# 9 1 0  
# 9 1 1  
# 10 0 0  
# 10 0 1  
# 10 1 0  
# 10 1 1  
# 11 0 0  
# 11 0 1  
# 11 1 0  
# 11 1 1  
# 12 0 0  
# 12 0 1  
# 12 1 0  
# 12 1 1  
# 13 0 0  
# 13 0 1  
# 13 1 0  
# 13 1 1  
# 14 0 0  
# 14 0 1  
# 14 1 0  
# 14 1 1  
# 15 0 0  
# 15 0 1  
# 15 1 0  
# 15 1 1  
# 16 0 0  
# 16 0 1  
# 16 1 0  
# 16 1 1  
# 17 0 0  
# 17 0 1  
# 17 1 0  
# 17 1 1  
# 18 0 0  
# 18 0 1  
# 18 1 0  
# 18 1 1  
# 19 0 0  
# 19 0 1  
# 19 1 0  
# 19 1 1  
# 20 0 0  
# 20 0 1  
# 20 1 0  
# 20 1 1  
# 21 0 0  
# 21 0 1  
# 21 1 0  
# 21 1 1  
# 22 0 0  
# 22 0 1  
# 22 1 0  
# 22 1 1  
# 23 0 0  
# 23 0 1  
# 23 1 0  
# 23 1 1  
# 24 0 0  
# 24 0 1  
# 24 1 0  
# 24 1 1  
# 25 0 0  
# 25 0 1  
# 25 1 0  
# 25 1 1  
# 26 0 0  
# 26 0 1  
# 26 1 0  
# 26 1 1  
# 27 0 0  
# 27 0 1  
# 27 1 0  
# 27 1 1  
# 28 0 0  
# 28 0 1  
# 28 1 0  
# 28 1 1  
# 29 0 0  
# 29 0 1  
# 29 1 0  
# 29 1 1  
# 30 0 0  
# 30 0 1  
# 30 1 0  
# 30 1 1  
# 31 0 0  
# 31 0 1  
# 31 1 0  
# 31 1 1  
# 32 0 0  
# 32 0 1  
# 32 1 0  
# 32 1 1  
# 33 0 0  
# 33 0 1  
# 33 1 0  
# 33 1 1  
# 34 0 0  
# 34 0 1  
# 34 1 0  
# 34 1 1  
# 35 0 0  
# 35 0 1  
# 35 1 0  
# 35 1 1  
# 36 0 0  
# 36 0 1  
# 36 1 0  
# 36 1 1  
# 37 0 0  
# 37 0 1  
# 37 1 0  
# 37 1 1  
# 38 0 0  
# 38 0 1  
# 38 1 0  
# 38 1 1  
# 39 0 0  
# 39 0 1  
# 39 1 0  
# 39 1 1  
# 40 0 0  
# 40 0 1  
# 40 1 0  
# 40 1 1  
# 41 0 0  
# 41 0 1  
# 41 1 0  
# 41 1 1  
# 42 0 0  
# 42 0 1  
# 42 1 0  
# 42 1 1  
# 43 0 0  
# 43 0 1  
# 43 1 0  
# 43 1 1  
# 44 0 0  
# 44 0 1  
# 44 1 0  
# 44 1 1  
# 45 0 0  
# 45 0 1  
# 45 1 0  
# 45 1 1  
# 46 0 0  
# 46 0 1  
# 46 1 0  
# 46 1 1  
# 47 0 0  
# 47 0 1  
# 47 1 0  
# 47 1 1  
# 48 0 0  
# 48 0 1  
# 48 1 0  
# 48 1 1  
# 49 0 0  
# 49 0 1  
# 49 1 0  
# 49 1 1  
# 50 0 0  
# 50 0 1  
# 50 1 0  
# 50 1 1  
# 51 0 0  
# 51 0 1  
# 51 1 0  
# 51 1 1  
# 52 0 0  
# 52 0 1  
# 52 1 0  
# 52 1 1  
# 53 0 0  
# 53 0 1  
# 53 1 0  
# 53 1 1  
# 54 0 0  
# 54 0 1  
# 54 1 0  
# 54 1 1  
# 55 0 0  
# 55 0 1  
# 55 1 0  
# 55 1 1  
# 56 0 0  
# 56 0 1  
# 56 1 0  
# 56 1 1  
# 57 0 0  
# 57 0 1  
# 57 1 0  
# 57 1 1  
# 58 0 0  
# 58 0 1  
# 58 1 0  
# 58 1 1  
# 59 0 0  
# 59 0 1  
# 59 1 0  
# 59 1 1  
# 60 0 0  
# 60 0 1  
# 60 1 0  
# 60 1 1  
# 61 0 0  
# 61 0 1  
# 61 1 0  
# 61 1 1  
# 62 0 0  
# 62 0 1  
# 62 1 0  
# 62 1 1  
# 63 0 0  
# 63 0 1  
# 63 1 0  
# 63 1 1  
# 64 0 0  
# 64 0 1  
# 64 1 0  
# 64 1 1  
# 65 0 0  
# 65 0 1  
# 65 1 0  
# 65 1 1  
# 66 0 0  
# 66 0 1  
# 66 1 0  
# 66 1 1  
# 67 0 0  
# 67 0 1  
# 67 1 0  
# 67 1 1  
# 68 0 0  
# 68 0 1  
# 68 1 0  
# 68 1 1  
# 69 0 0  
# 69 0 1  
# 69 1 0  
# 69 1 1  
# 70 0 0  
# 70 0 1  
# 70 1 0  
# 70 1 1  
# 71 0 0  
# 71 0 1  
# 71 1 0  
# 71 1 1  
# 72 0 0  
# 72 0 1  
# 72 1 0  
# 72 1 1  
# 73 0 0  
# 73 0 1  
# 73 1 0  
# 73 1 1  
# 74 0 0  
# 74 0 1  
# 74 1 0  
# 74 1 1  
# 75 0 0  
# 75 0 1  
# 75 1 0  
# 75 1 1  
# 76 0 0  
# 76 0 1  
# 76 1 0  
# 76 1 1  
# 77 0 0  
# 77 0 1  
# 77 1 0  
# 77 1 1  
# 78 0 0  
# 78 0 1  
# 78 1 0  
# 78 1 1  
# 79 0 0  
# 79 0 1  
# 79 1 0  
# 79 1 1  
# 80 0 0  
# 80 0 1  
# 80 1 0  
# 80 1 1  
# 81 0 0  
# 81 0 1  
# 81 1 0  
# 81 1 1  
# 82 0 0  
# 82 0 1  
# 82 1 0  
# 82 1 1  
# 83 0 0  
# 83 0 1  
# 83 1 0  
# 83 1 1  
# 84 0 0  
# 84 0 1  
# 84 1 0  
# 84 1 1  
# 85 0 0  
# 85 0 1  
# 85 1 0  
# 85 1 1  
# 86 0 0  
# 86 0 1  
# 86 1 0  
# 86 1 1  
# 87 0 0  
# 87 0 1  
# 87 1 0  
# 87 1 1  
# 88 0 0  
# 88 0 1  
# 88 1 0  
# 88 1 1  
# 89 0 0  
# 89 0 1  
# 89 1 0  
# 89 1 1  
# 90 0 0  
# 90 0 1  
# 90 1 0  
# 90 1 1  
# 91 0 0  
# 91 0 1  
# 91 1 0  
# 91 1 1  
# 92 0 0  
# 92 0 1  
# 92 1 0  
# 92 1 1  
# 93 0 0  
# 93 0 1  
# 93 1 0  
# 93 1 1  
# 94 0 0  
# 94 0 1  
# 94 1 0  
# 94 1 1  
# 95 0 0  
# 95 0 1  
# 95 1 0  
# 95 1 1  
# 96 0 0  
# 96 0 1  
# 96 1 0  
# 96 1 1  
# 97 0 0  
# 97 0 1  
# 97 1 0  
# 97 1 1  
# 98 0 0  
# 98 0 1  
# 98 1 0  
# 98 1 1  
# 99 0 0  
# 99 0 1  
# 99 1 0  
# 99 1 1  
# 100 0 0  
# 100 0 1  
# 100 1 0  
# 100 1 1
```

```
* Return int the number of the sector (0 if x =  
static function get_sector ($x, $y, $z) {
```









```
* Common fields
Platform = Platform[0]
* placeholders
* placeholders
keytype = ""
```

```
* The coordinates (0, 0, 0) represents the octocube
*/
class GeoOctocube {
```

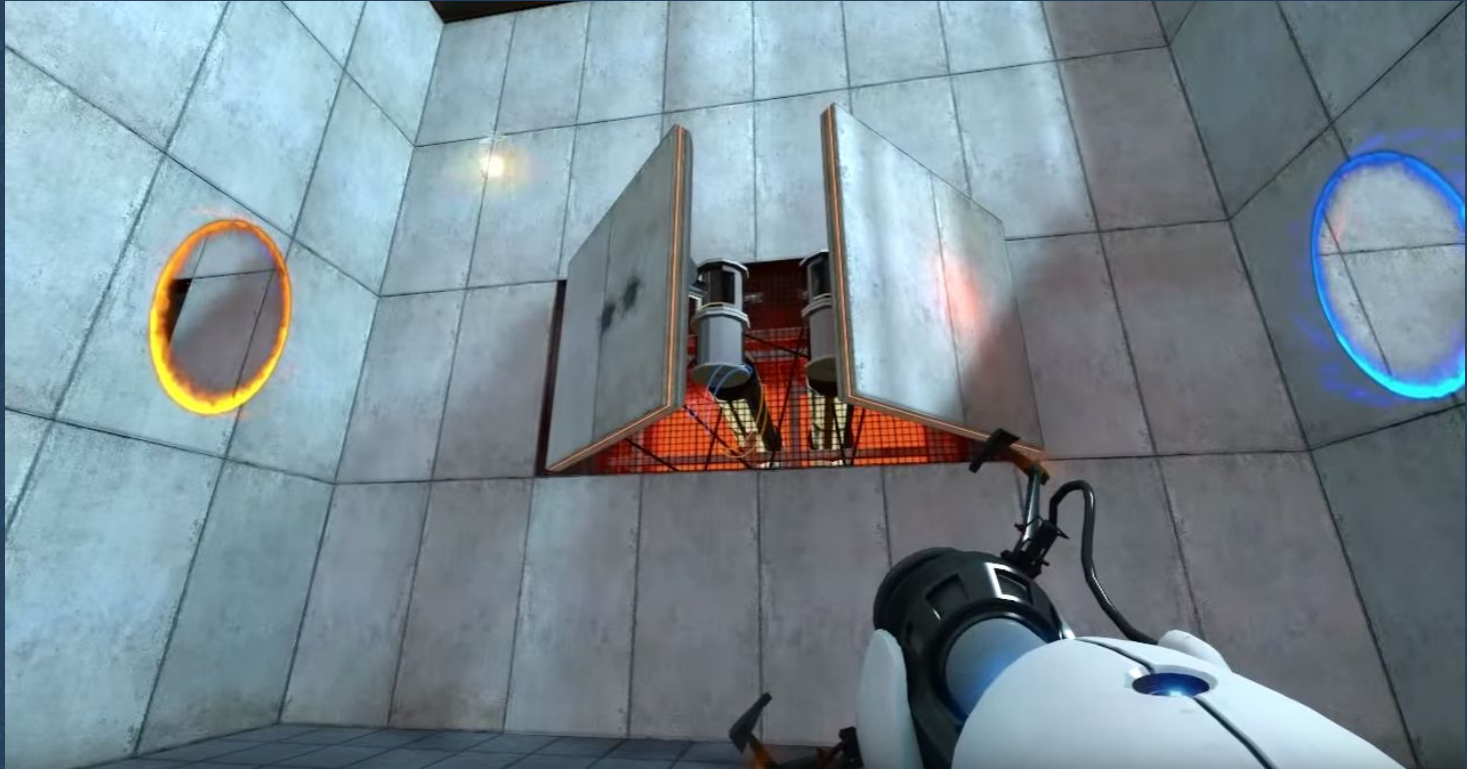
specified

```
require 'base32'
require 'cryptosol'
```

```
module Crypto
  * Blocks are
  * Cryptosphere
  sauce, well
  Blocks provide
  between 0 bytes
  convergent e
  approach, a cry
  together with a
  symmetric key
  authenticated sym
  pre specifics
  lock
  key to use wh
  matches the U
  = "crypt_block
  it on the size of
  = 1_048_576
```

```
blake2bXSalsa20pp
  * @return int the number of the sector (0 if x =
  static function get_sector ($x, $y, $z) {
```



















 Death

- Conjure Death by pressing 
- Aim with  to shoot a beam!
- Kill those pesky Goblins!

Tovlca, Shintaman



# Revisit/reuse areas



```
* The coordinates (0, 0, 0) represents the octocube
*/
class GeoOctocube {
    /*
    * Gets the sector from the (x, y, z) specified
    *
    * Sector will be:
    *
    * @return int the number of the sector (0 if x =
    static function get_sector($x, $y, $z) {
```



require 'base32'  
require 'cryptosphere' # Blocks are the underlying Cryptosphere for data communication. Blocks provide for both data convergent encryption technique, a cryptographic hash approach, a symmetric key is derived, which authenticated symmetric cipher. More specifics on the encryption key to use when calculating the ID of a block matches the URI scheme for blocks = "crypt\_block"

```
class GeoOctocube {  
  * The coordinates (0, 0, 0) represents the octocube  
  */  
  static function get_sector ($x, $y, $z) {  
    @return int the number of the sector (0 if x =  
    $x < 0, y < 0, z < 0)  
  }  
}
```

the (x, y, z) specified

coordinate  
coordinate  
coordinate

see Blake2bXSalsa20p

static function get\_sector (\$x, \$y, \$z) {

# In Nintendo games:

- Concept
- Development
- Twist
- Conclusion

```
* The coordinates (0, 0, 0) represents the octocube
*/
class GeoOctocube {
```

\* Gets the sector from the (x, y, z) specified

\* Sector will be:

\* <code>



\* @param int \$x the x coordinate  
\* @param int \$y the y coordinate  
\* @param int \$z the z coordinate

\* @return int the number of the sector (0 if x =

```
static function get_sector ($x, $y, $z) {
```



# Alternatively...

- Introduce ability 1
  - Let player use it
- Introduce ability 2
  - Let player use it
- Combine 1 + 2
  - Safe environment
  - Under pressure
- Introduce ability 3...

```
* The coordinates (0, 0, 0) represents the octocube
*/
class GeoOctocube {
    /**
     * Gets the sector from the (x, y, z) specified
     *
     * Sector will be:
     * <code>
```



```
* @param int $x the x coordinate
* @param int $y the y coordinate
* @param int $z the z coordinate
```

```
* @return int the number of the sector (0 if x =
static function get_sector($x, $y, $z) {
```



# Remember...

Let them try.

Let them fail.

Let them learn.

Let them play.

```

* Common fields
format = fields[0]
paddingchars = [] * placeholder
keytype = "" * placeholder

```

```

* key heuristic to distinguish known_hosts from known_hosts2.
* ... field entirely decimal digits?
... fields[1]

```

```

* Treat all fields as host key
* Format: hostkey [ssh] [ssh] comment
* (PUTT doesn't store the type of bits)
paddingchars = map (long, HostKey)
padding = ""

```

```

* The coordinates (0, 0, 0) represents the octocube
*/
class GeoOctocube {

```

```

* Gets the sector from the (x, y, z) specified

```

```

* Sector will be:

```

	5	6	
-	-	-	-
7	8		2
3	4		

```

* <code>

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

* //

```

```

static function get_sector ($x, $y, $z) {

```

```

function() {return ""}
public function __construct($x, $y, $z) {
    $this->x = $x;
    $this->y = $y;
    $this->z = $z;
}
public function __get($property) {
    $value = null;
    if (isset($this->{$property})) {
        $value = $this->{$property};
    }
    return $value;
}
public function __set($property, $value) {
    $this->{$property} = $value;
}
public function __isset($property) {
    return isset($this->{$property});
}
public function __unset($property) {
    unset($this->{$property});
}
public function __clone() {
    $this->x = $this->x;
    $this->y = $this->y;
    $this->z = $this->z;
}
public function __toString() {
    return "GeoOctocube($x, $y, $z)";
}

```

# Juiciness

```
* Common fields
*
* @param $fields[0]
* @param $fields[1]
* @param $fields[2]
* @param $fields[3]
```

```
* Drotty heuristic to distinguish known hosts from known hosts2.
*
* @param $fields[0]
* @param $fields[1]
* @param $fields[2]
* @param $fields[3]
```

```
* Travel all fields, type host key
*
* @param $fields[0]
* @param $fields[1]
* @param $fields[2]
* @param $fields[3]
```

```
* Format hostkey as a string of bits
*
* @param $fields[0]
* @param $fields[1]
* @param $fields[2]
* @param $fields[3]
```

```
* (PUTT doesn't store the size of bits)
*
* @param $fields[0]
* @param $fields[1]
* @param $fields[2]
* @param $fields[3]
```

```
* regionname = map (long, float)
*
* @param $fields[0]
* @param $fields[1]
* @param $fields[2]
* @param $fields[3]
```

```
* The coordinates (0, 0, 0) represents the octocube
*/
class GeoOctocube {
```

\* Gets the sector from the (x, y, z) specified

\* Sector will be:

\* <code>

		5	6		
		7	8		2
		3	4		

\* @param int \$x the x coordinate  
\* @param int \$y the y coordinate  
\* @param int \$z the z coordinate

\* @return int the number of the sector (0 if x =

```
static function get_sector ($x, $y, $z) {
```

```
function get_sector ($x, $y, $z) {
    $x = floor($x/5);
    $y = floor($y/5);
    $z = floor($z/5);
    $sector = 0;
    for ($i = 0; $i < 5; $i++) {
        for ($j = 0; $j < 5; $j++) {
            for ($k = 0; $k < 5; $k++) {
                $sector++;
            }
        }
    }
    return $sector;
}
```





# What is it good for?

- Providing feedback
- Making the game feel “alive”
- Worldbuilding
- Visibility of System Status

```
* The coordinates (0, 0, 0) represents the octocube
*/
class GeoOctocube {
```

\* Gets the sector from the (x, y, z) specified

\* Sector will be:

\* `<code>`



\* param int \$x the x coordinate  
\* param int \$y the y coordinate  
\* param int \$z the z coordinate

\* @return int the number of the sector (0 if x =

```
static function get_sector ($x, $y, $z) {
```







# Interface...



vs.



# Interface <3 Gameplay





# Recap

- Usability
  - The challenge in the gameplay, not in navigation
- Tutorials
  - Teach the player your mechanics
- Juiciness = audio/visual feedback
  - Design for this
  - Provides feedback
  - Polish
  - Worldbuilding
  - Visibility of System Status
- More on juiciness in gameplay:
  - <https://www.youtube.com/watch?v=Fy0aCDmgngxg>

