

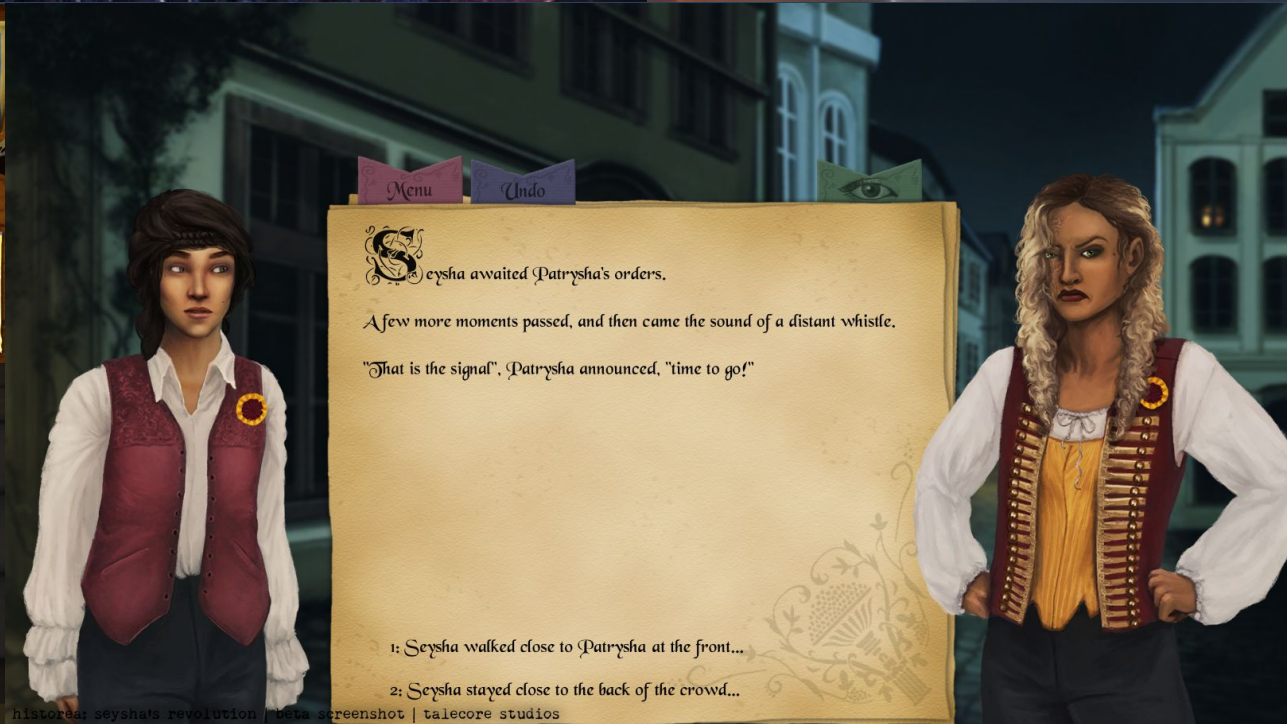
# UX & interaction design

## Intro



# Who am I?

- Anna Jenelius
- Bachelor's Degree in Game Development (Stockholm University)
- First game development job in 2011 (Animation)
- Started at Paradox in 2012
  - QA Tester
  - Assistant QA Manager
  - Senior QA Manager
- 2015, started Talecore Studios (indie dev, freelancer, teacher)
- Now: Founder, CEO & Creative Director at Valiant Game Studio
- Also: Organizing gamedev meetups (Link in Park/Link in Bar), etc





# PENDULA SWITC



# Q: What does UX stand for?

```
/* Common fields
 * Format: hostip[0]
 * regionnumber = []
 * hosttype = ""
 * placeholder

 * Dotted heuristic to distinguish known hosts from known hosts2.
 * Is second field entirely decimal digits?
 * @param ip[0] hostip[0] fields[1]
 * @return bool true if host key
 * @format hostip[0] regionnumber[0]
 * @PUTTY doesn't store the order of bits
 * regionnumber = map (long, HostType)
 * hosttype = ""
 */
```

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

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

\* Sector will be:

\* <code>



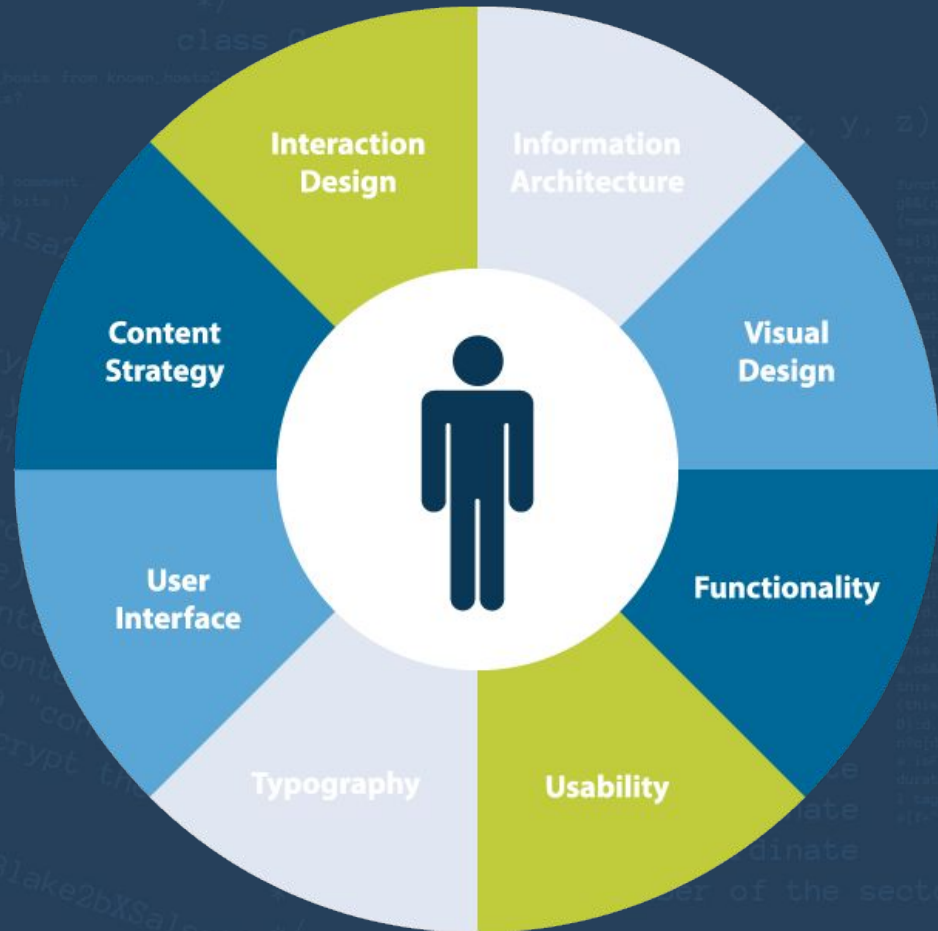
\* @param int \$x the x coordinate  
 \* @param int \$y the y coordinate  
 \* @param int \$z the z coordinate  
 \* @return int the number of the sector (0 if x =

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

```
function get_sector($x, $y, $z) {
    $x = floor($x/2);
    $y = floor($y/2);
    $z = floor($z/2);
    $sector = 0;
    if ($x < 0) {
        $sector = 1;
    }
    if ($y < 0) {
        $sector = 2;
    }
    if ($z < 0) {
        $sector = 3;
    }
    if ($x > 0) {
        $sector = 4;
    }
    if ($y > 0) {
        $sector = 5;
    }
    if ($z > 0) {
        $sector = 6;
    }
    return $sector;
}
```

Answer:

User Experience







# Purpose

- Display information
- Interact with game

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

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

\* Sector will be:



```
* @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 kinds of UIs?

```

* Common fields
postal = fields[0]
regionnumbers = []
* placeholder
* placeholder
keytype = ""

```

```

* Dirty heuristic to distinguish known_hosts from known_hosts2:
is second field entirely decimal digits?
return (int)fields[1]
* Travel all the way to host key
* Format hostkey with explicit width comment
* (PUTTY doesn't store the size of bits)
regionnumbers = map(lambda field: int(field), fields[1])
regionnum = " "

```

```

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

```

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

\* Sector will be:



\* @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() {return $x+$y+$z;
public function __construct($x, $y, $z) {
    $this->x = $x;
    $this->y = $y;
    $this->z = $z;
}
public function get_sector() {
    $sector = floor($x/5)*6 + floor($y/5)*4 + floor($z/5)*2;
    return $sector;
}
}

```

# Menus

- Main menu
  - New game
  - Load save/Continue
  - Settings
  - Credits
  - Exit
  - Social buttons



# Menus

- Pause menu
  - Save
  - Load save
  - Stats
  - Settings
  - Exit to menu
  - Exit to desktop



# HUD



# HUD



# Inventory



# Codex/logs







# Terminology

**Diegetic:** Interface that is included in the game world -- i.e., it can be seen and heard by the game characters.

**Non-diegetic:** Interface that is rendered outside the game world, only visible and audible to the players in the real world.

**Spatial:** UI elements presented in the game's 3D space with or without being an entity of the actual game world (diegetic or non-diegetic).

**Meta:** Representations can exist in the game world, but aren't necessarily visualized spatially for the player; these are **meta representations**.

# Diegetic



# Diegetic



# Non-Diegetic



# Non-Diegetic



# Spatial

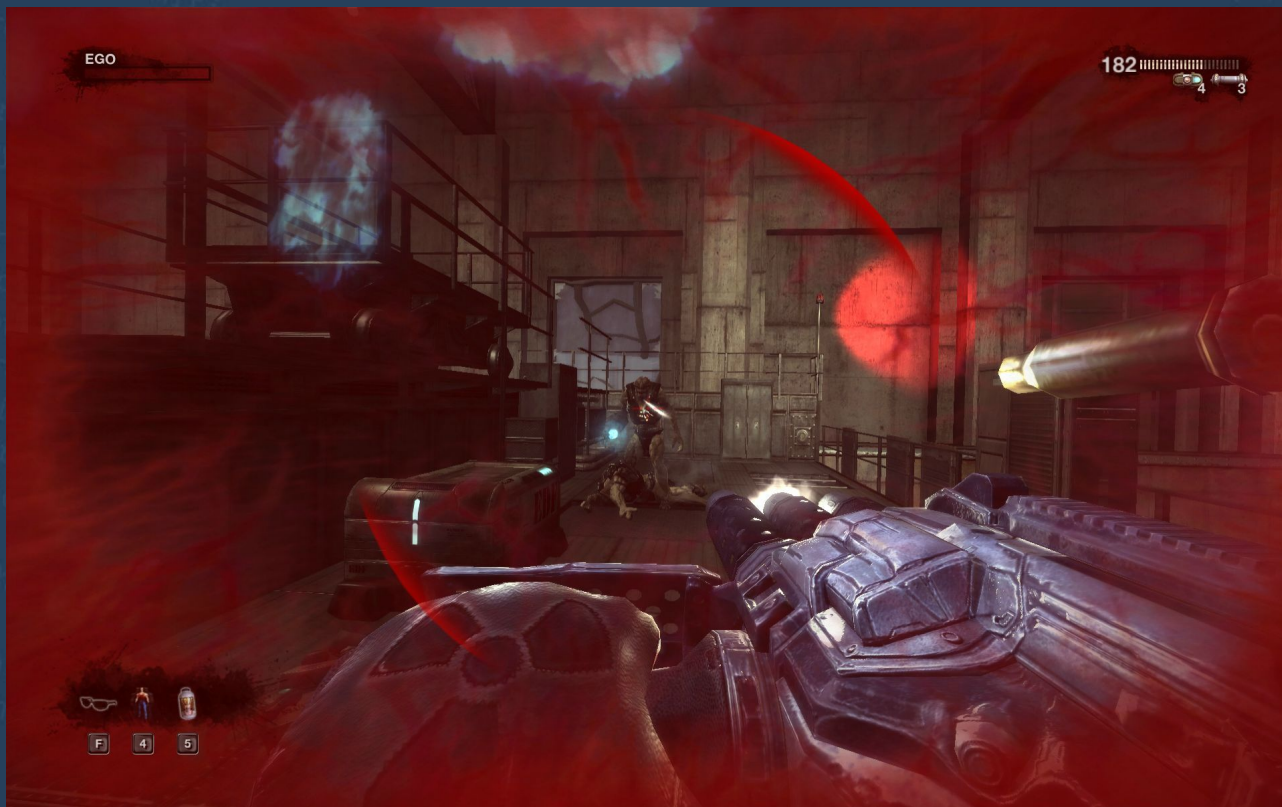


# Spatial





# Meta



```
# require 'base32'
require 'cryptosphere'
module Cryptosphere
  # Blocks are the
  # sauce, welcome m
  # Blocks provide for
  # between 0 bytes and
  # convergent encrypt
  # approach, a cryptogr
  # together with an opti
  # symmetric key is de
  # authenticated symmetri
  # more specifics on bl
  # key to use when ca
  # matches the URI so
  # = "crypt_block"
  # it on the size of wh
  # = 1_048_576
  # * Common fields
  # format = fields[0]
  # maximum_size = 1 # placeholder
  # key_type = "" # placeholder
  # * Dotted heuristic to distinguish known_hosts from known_hosts2.
  # * second field entirely decimal digit?
  # @param [String] fields[1]
  # The coordinates (0, 0, 0) represents the octocube
  */
  class GeoOctocube {
    # Gets the sector from the (x, y, z) specified
    def self.get_sector(x, y, z)
      # ...
    end
  end
end
```

# Meta



**Is the representation visualized in the 3D game space?**

no                      yes

no	<b>non-diegetic representations</b>	<b>spatial representations</b>
yes	<b>meta representations</b>	<b>diegetic representations</b>

**Is the representation existing in the fictional game world?**

no

yes

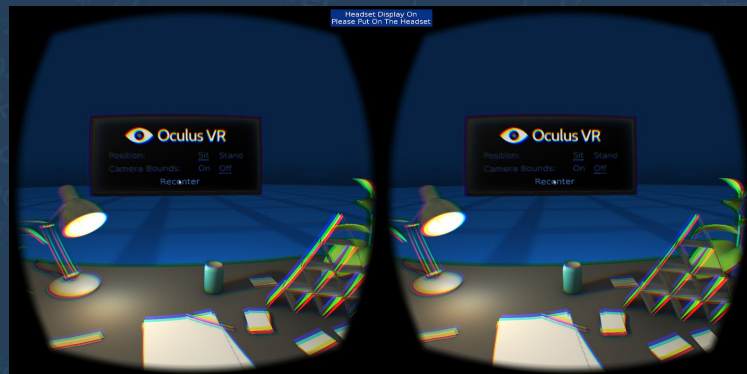
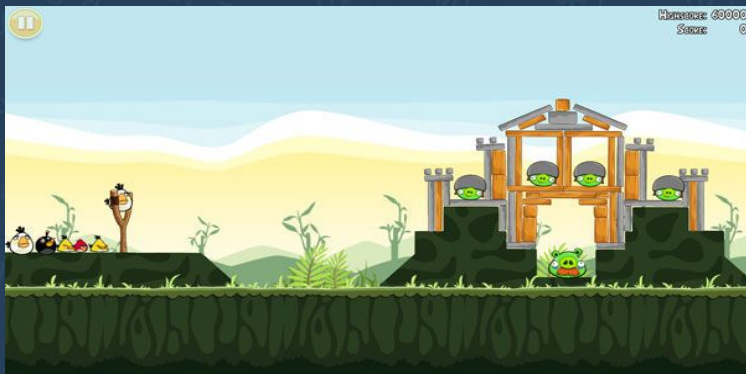
```

        direction = direction + 1
        if direction == 0:
            x = x + 1
        elif direction == 1:
            y = y + 1
        elif direction == 2:
            z = z + 1
        elif direction == 3:
            x = x - 1
        elif direction == 4:
            y = y - 1
        elif direction == 5:
            z = z - 1
        return (x, y, z)

    def get_sector(self, x, y, z):
        """Returns the sector number for the given coordinates.
        The sectors are numbered from 0 to 7.
        The sectors are arranged in a circle around the center.
        The sectors are numbered as follows:
        0: North
        1: North-East
        2: East
        3: South-East
        4: South
        5: South-West
        6: West
        7: North-West
        """
        # Get the center from the (x, y, z) specified
        center = self.get_center(x, y, z)
        # Get the direction from the (x, y, z) specified
        direction = self.get_direction(x, y, z, center)
        # Get the sector from the (x, y, z) specified
        sector = self.get_sector(x, y, z)
        return sector
    
```



# Platform specifics





# PC vs. console



# PC vs. console





# PC vs. console



# Mobile

- Big elements
- No hover
- Consider “handedness” of player

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

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

\* Sector will be:



\* 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) {
```

# Mobile



# VR



```
* The coordinates (0, 0, 0) represents the octocube
*/
class GeoOctocube {
    * Gets the sector from the (x, y, z) specified
    *
    function getSector($x, $y, $z) {
        $x = (int)$x;
        $y = (int)$y;
        $z = (int)$z;
        // ...
    }
}

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

static function getSector($x, $y, $z) {
    // ...
}

```

# VR

- Hard having things stuck to camera
- Text can easily become pixelated
- Diegetic UI works well

```
* 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

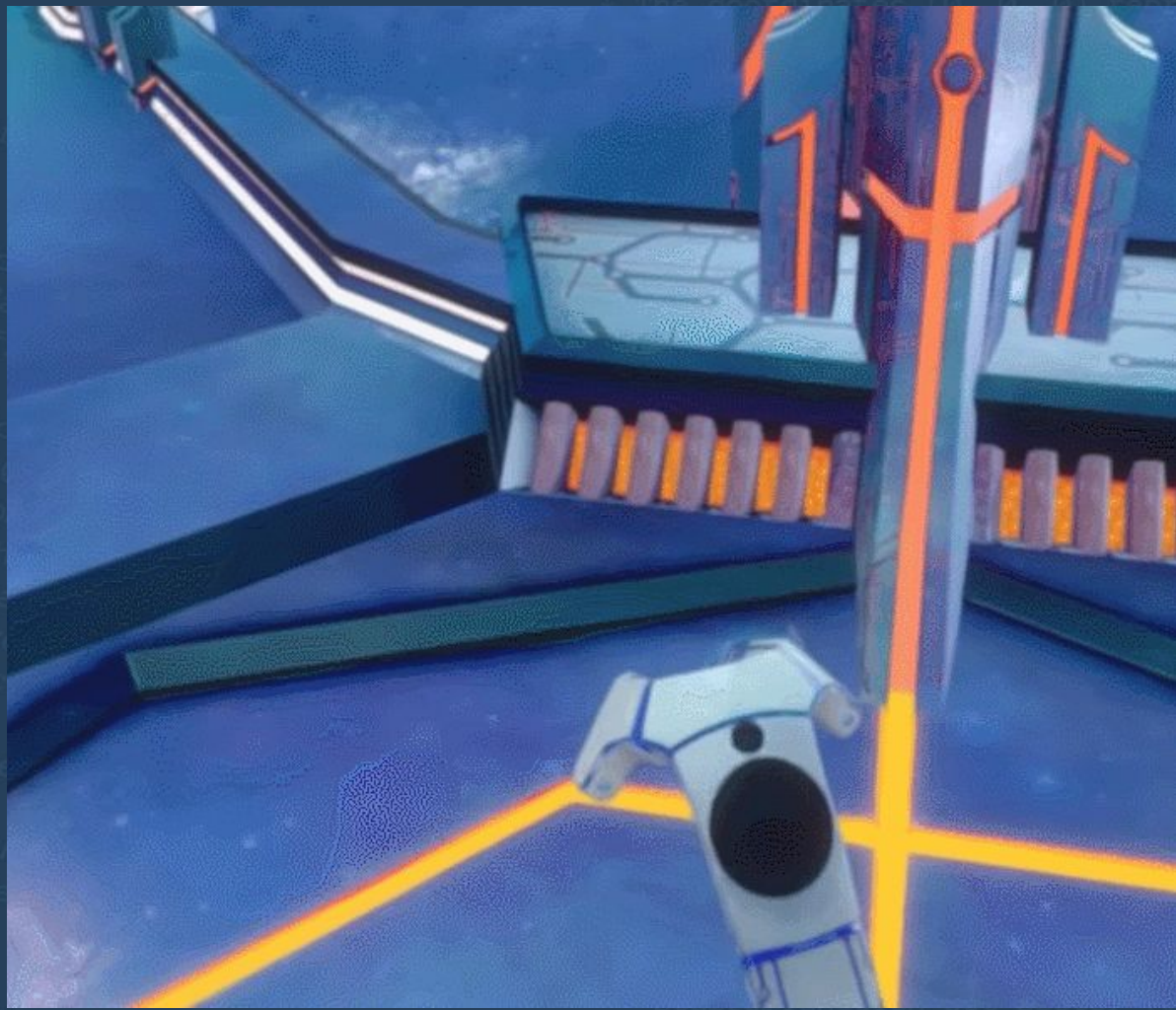
\* @return int the number of the sector (0 if x =

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





2







from the (x, y, z) specified



x coordinate  
y coordinate  
z coordinate

number of the sector (0 if x =

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



# Immersion

- Give the info
- Don't take the player "out" of the experience
- Don't be intrusive

```
* 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) {
```

# Make important things obvious

- Don't have to see to "get"
  - Damage - red edges
  - Velocity in car game



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

# KISS (Keep It Simple, Stupid)

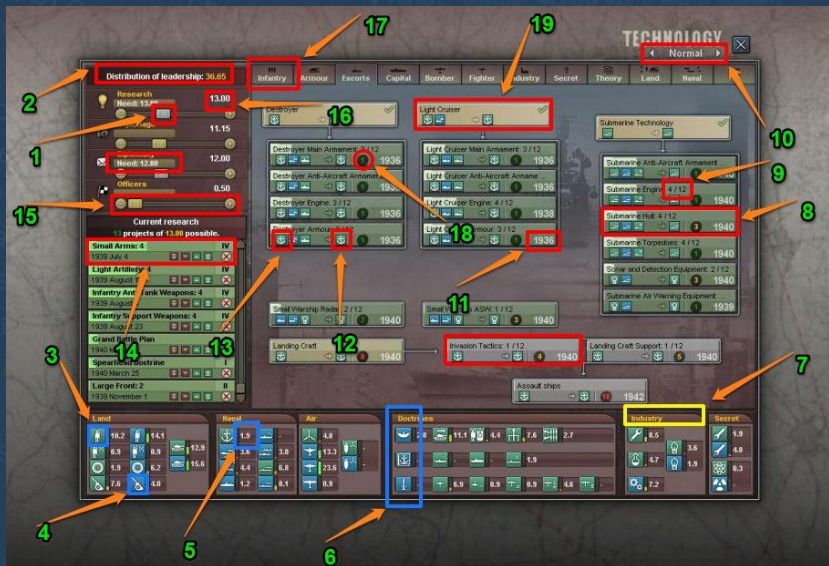
The screenshot displays a shipping software interface with the following sections:

- Order Header:** Order # 93004234, Date 09/03/927, Status New, Request Selection.
- Origin/To:** From SC To SC, Mode Air, Rate 100670661.
- Service:** CAVRS-00-01, Service 0194.
- Delivery:** Deliver By 06-12-02 17:00, Clock Step.
- Origin Address:** VANCOUVER, BC V6Z2K8, Ph: [Redacted], Cont: [Redacted], Appointment D: 06-10-02.
- Destination Address:** CANADIAN HARDWARE & H, AVENUE SUITE 101, SCARBOROUGH, ON M1B5M4, Ph: [Redacted], Cont: [Redacted], Appointment D: [Redacted].
- Charges Summary:**

Charges:	761.50
Discount:	0%
Sub Total:	761.50
Accessorial:	40.00
DV:	0.00
FSC, Act:	2.50% 38.00
Total:	839.50
Balance:	839.50
- Units Table:**

Units	Type	H. Description	Stated	ASMT	Dimensions (LxWxH)	CHW	Rate	Charge	
1	CRATE	CRATE	91	94	97	55x50x30	97	50.00	40.50
1	CRATE	1 MAN FLD	508		1,426	50x40x40	1,426	50.00	713.00
0								0.00	0.00
- Summary:** 1 Accs, \$40.00, DV: 0, \$0.00, 591, 94, 152, 1,523, 761.50

# Layer your interfaces



# Visibility of System Status

“The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.”

- Nielsen, 1995



# Recap

- UI = User Interface
- There are several different kinds, different purposes
  - In/outside the game world
  - Visible/Invisible to the characters
- Different platforms have different pros and cons
- Basic design practices
  - Keeping immersion
  - Don't intrude
  - Layer your interfaces
  - Visibility of system status



```
static inline int  
get_id_sector(int x, int y, int z)  
{  
    int id; // id of the sector  
    int x2, y2, z2; // coordinates of the sector  
    int x1, y1, z1; // coordinates of the sector  
    int x3, y3, z3; // coordinates of the sector  
    int x4, y4, z4; // coordinates of the sector  
    int x5, y5, z5; // coordinates of the sector  
    int x6, y6, z6; // coordinates of the sector  
    int x7, y7, z7; // coordinates of the sector  
    int x8, y8, z8; // coordinates of the sector  
    int x9, y9, z9; // coordinates of the sector  
    int x10, y10, z10; // coordinates of the sector  
    int x11, y11, z11; // coordinates of the sector  
    int x12, y12, z12; // coordinates of the sector  
    int x13, y13, z13; // coordinates of the sector  
    int x14, y14, z14; // coordinates of the sector  
    int x15, y15, z15; // coordinates of the sector  
    int x16, y16, z16; // coordinates of the sector  
    int x17, y17, z17; // coordinates of the sector  
    int x18, y18, z18; // coordinates of the sector  
    int x19, y19, z19; // coordinates of the sector  
    int x20, y20, z20; // coordinates of the sector  
    int x21, y21, z21; // coordinates of the sector  
    int x22, y22, z22; // coordinates of the sector  
    int x23, y23, z23; // coordinates of the sector  
    int x24, y24, z24; // coordinates of the sector  
    int x25, y25, z25; // coordinates of the sector  
    int x26, y26, z26; // coordinates of the sector  
    int x27, y27, z27; // coordinates of the sector  
    int x28, y28, z28; // coordinates of the sector  
    int x29, y29, z29; // coordinates of the sector  
    int x30, y30, z30; // coordinates of the sector  
    int x31, y31, z31; // coordinates of the sector  
    int x32, y32, z32; // coordinates of the sector  
    int x33, y33, z33; // coordinates of the sector  
    int x34, y34, z34; // coordinates of the sector  
    int x35, y35, z35; // coordinates of the sector  
    int x36, y36, z36; // coordinates of the sector  
    int x37, y37, z37; // coordinates of the sector  
    int x38, y38, z38; // coordinates of the sector  
    int x39, y39, z39; // coordinates of the sector  
    int x40, y40, z40; // coordinates of the sector  
    int x41, y41, z41; // coordinates of the sector  
    int x42, y42, z42; // coordinates of the sector  
    int x43, y43, z43; // coordinates of the sector  
    int x44, y44, z44; // coordinates of the sector  
    int x45, y45, z45; // coordinates of the sector  
    int x46, y46, z46; // coordinates of the sector  
    int x47, y47, z47; // coordinates of the sector  
    int x48, y48, z48; // coordinates of the sector  
    int x49, y49, z49; // coordinates of the sector  
    int x50, y50, z50; // coordinates of the sector  
    int x51, y51, z51; // coordinates of the sector  
    int x52, y52, z52; // coordinates of the sector  
    int x53, y53, z53; // coordinates of the sector  
    int x54, y54, z54; // coordinates of the sector  
    int x55, y55, z55; // coordinates of the sector  
    int x56, y56, z56; // coordinates of the sector  
    int x57, y57, z57; // coordinates of the sector  
    int x58, y58, z58; // coordinates of the sector  
    int x59, y59, z59; // coordinates of the sector  
    int x60, y60, z60; // coordinates of the sector  
    int x61, y61, z61; // coordinates of the sector  
    int x62, y62, z62; // coordinates of the sector  
    int x63, y63, z63; // coordinates of the sector  
    int x64, y64, z64; // coordinates of the sector  
    int x65, y65, z65; // coordinates of the sector  
    int x66, y66, z66; // coordinates of the sector  
    int x67, y67, z67; // coordinates of the sector  
    int x68, y68, z68; // coordinates of the sector  
    int x69, y69, z69; // coordinates of the sector  
    int x70, y70, z70; // coordinates of the sector  
    int x71, y71, z71; // coordinates of the sector  
    int x72, y72, z72; // coordinates of the sector  
    int x73, y73, z73; // coordinates of the sector  
    int x74, y74, z74; // coordinates of the sector  
    int x75, y75, z75; // coordinates of the sector  
    int x76, y76, z76; // coordinates of the sector  
    int x77, y77, z77; // coordinates of the sector  
    int x78, y78, z78; // coordinates of the sector  
    int x79, y79, z79; // coordinates of the sector  
    int x80, y80, z80; // coordinates of the sector  
    int x81, y81, z81; // coordinates of the sector  
    int x82, y82, z82; // coordinates of the sector  
    int x83, y83, z83; // coordinates of the sector  
    int x84, y84, z84; // coordinates of the sector  
    int x85, y85, z85; // coordinates of the sector  
    int x86, y86, z86; // coordinates of the sector  
    int x87, y87, z87; // coordinates of the sector  
    int x88, y88, z88; // coordinates of the sector  
    int x89, y89, z89; // coordinates of the sector  
    int x90, y90, z90; // coordinates of the sector  
    int x91, y91, z91; // coordinates of the sector  
    int x92, y92, z92; // coordinates of the sector  
    int x93, y93, z93; // coordinates of the sector  
    int x94, y94, z94; // coordinates of the sector  
    int x95, y95, z95; // coordinates of the sector  
    int x96, y96, z96; // coordinates of the sector  
    int x97, y97, z97; // coordinates of the sector  
    int x98, y98, z98; // coordinates of the sector  
    int x99, y99, z99; // coordinates of the sector  
    int x100, y100, z100; // coordinates of the sector  
    return id;  
}
```



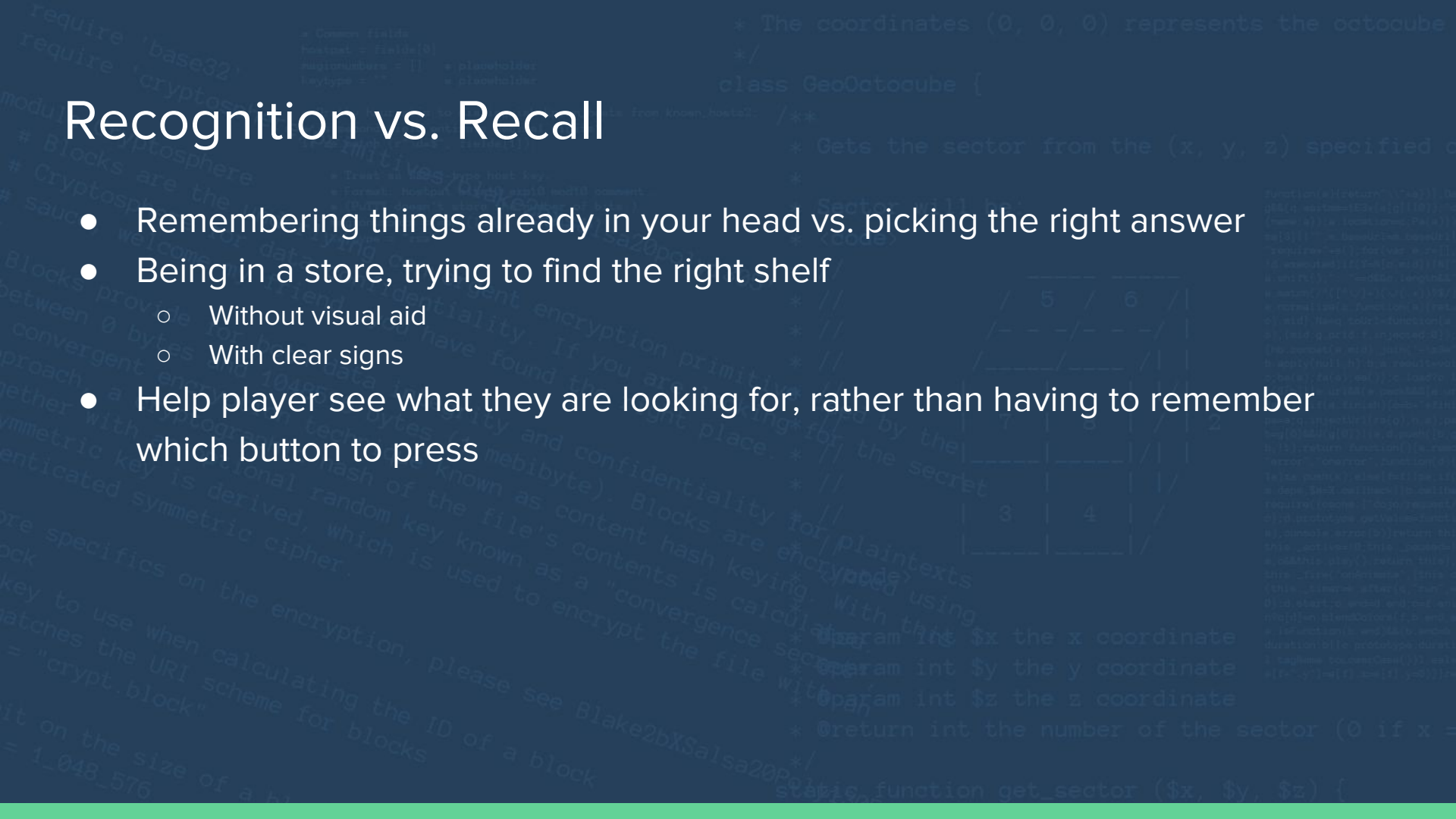
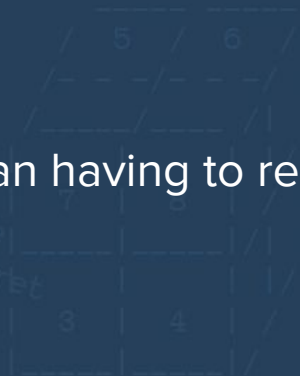






# Recognition vs. Recall

- Remembering things already in your head vs. picking the right answer
- Being in a store, trying to find the right shelf
  - Without visual aid
  - With clear signs
- Help player see what they are looking for, rather than having to remember which button to press

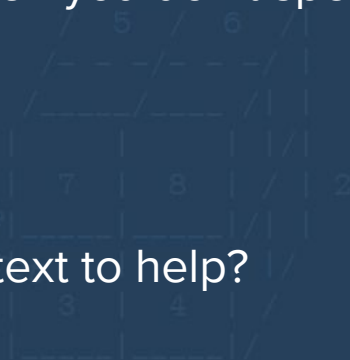


# An example:

If your little sister changes the language of your phone to Russian, would you find your way back to the right menu and change it back (given you don't speak Russian?)

Why/Why not?

How can you design a UI that you can navigate without text to help?



3G

9:42 AM



**В В С** РУССКАЯ СЛУЖБА

Главные новости

Последние новости

## У мэрии Москвы произошла стычка из-за митингов



Сторонники и противники российского премьер-министра Владимира Путина устроили потасовки у...

## Медведев встретится с лидерами неразрешенных партий



Уходящий президент России Дмитрий Медведев в понедельник, как ожидается, проведет редкую в...

## Новым президентом Германии станет правозащитник из ГДР



Канцлер Германии Ангела Меркель заявила, что поддерживает кандидатуру



Новости



Популярное



Разделы



Настройки

# Common Imagery









# Affordance

```
* Common fields
*
* @param $fields[0]
* @param $regions[0]
* @param $keytype
```

```
* Drotty heuristic to distinguish known_hosts from known_hosts2.
*
* @param $fields[0]
* @param $fields[1]
```

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

```
* 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) {
```

```
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;
}
```

# How to interact with an object



# Mimic physical items





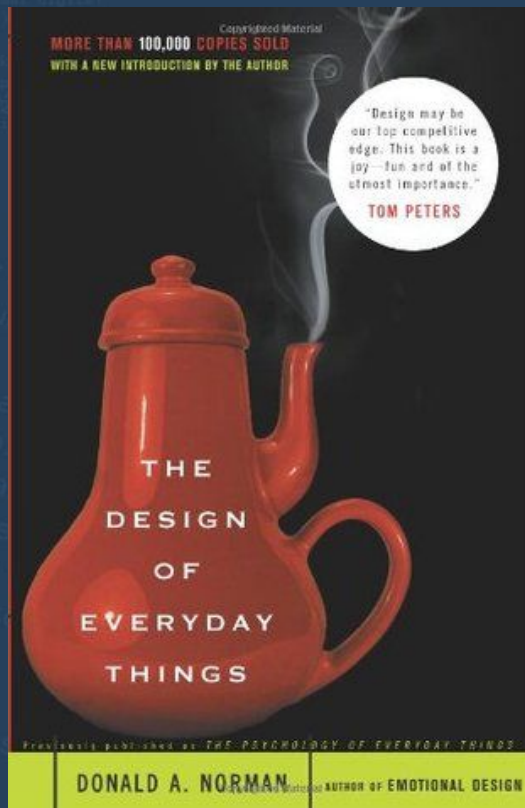
```
require 'base32'
require 'cryptosphere'

module Cryptosphere
  # Blocks are the underlying data
  # Cryptosphere for data
  # sauce, welcome my friend

  # Blocks provide for both
  # between 0 bytes and 16
  # convergent encryption
  # approach, a cryptographic
  # together with an optional
  # symmetric key is derived
  # authenticated symmetric
  # more specifics on the
  # key to use when calculating the ID of a block
  # matches the URI scheme for blocks
  = "crypt_block"
  # it on the size of a
  = 1_048_576

  # The coordinates (0, 0, 0) represents the octocube
  */
  class GeoOctocube {
    /**
     * Gets the sector from the (x, y, z) specified
     * @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) {
  
```

# Reading tip:



# Buttons

```
* 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]
```

```
* Drotty heuristic to distinguish known hosts from known hosts2.
*
* @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
```

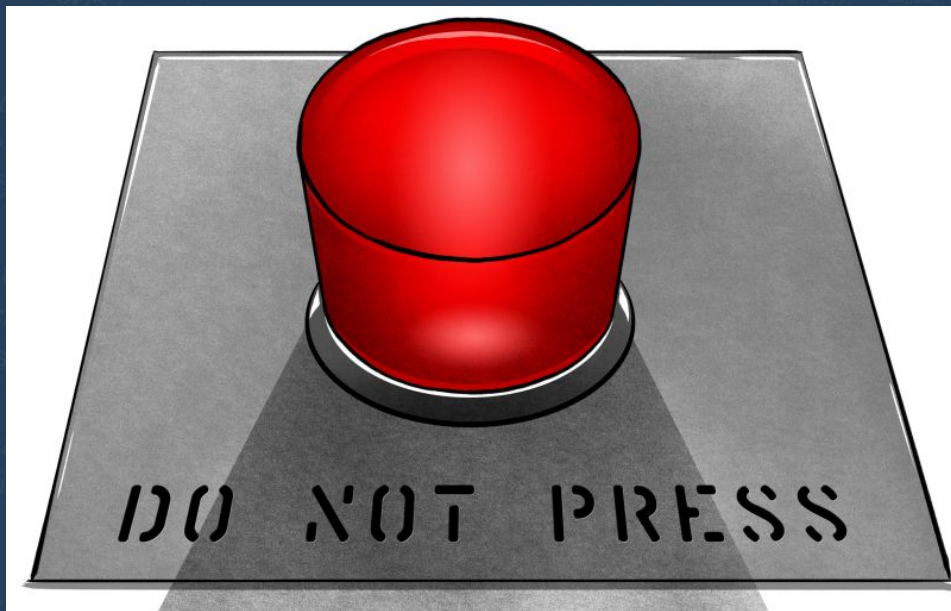
```
* @return int the number of the sector (0 if x =
```

```
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);
    $sector = 0;
    if ($x < 0) {
        $sector = 1;
    }
    if ($y < 0) {
        $sector = 2;
    }
    if ($z < 0) {
        $sector = 3;
    }
    if ($x > 0) {
        $sector = 4;
    }
    if ($y > 0) {
        $sector = 5;
    }
    if ($z > 0) {
        $sector = 6;
    }
    return $sector;
}
```

# Buttons

- Protrude from surroundings
- Are pushed
- Binary (ON/OFF)





# For buttons...

- Make them "3D"
  - Shadows
  - Highlights
  - Bevel
- Effect when is pushed
- Possibly highlight effect
- Show current status (ON/OFF)
- Gray out when disabled

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

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

\* Sector will be:  
\* <code>

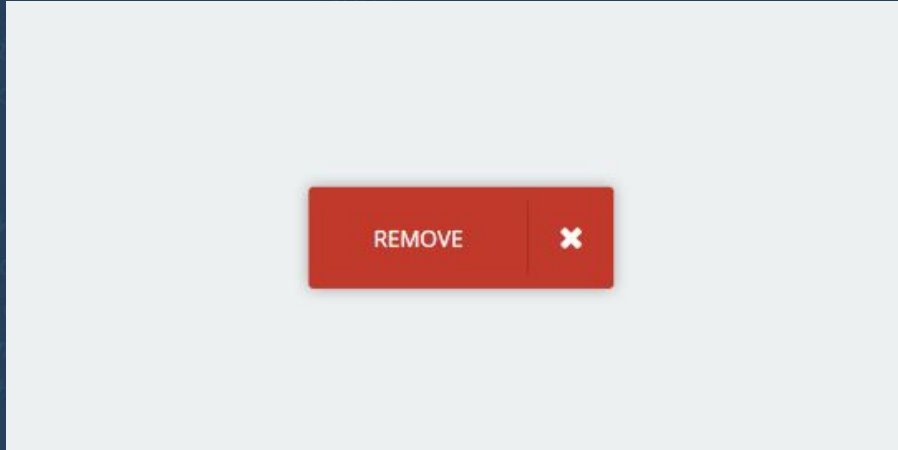


\* @param int \$x the x coordinate  
\* @param int \$y the y coordinate  
\* @param int \$z the z coordinate

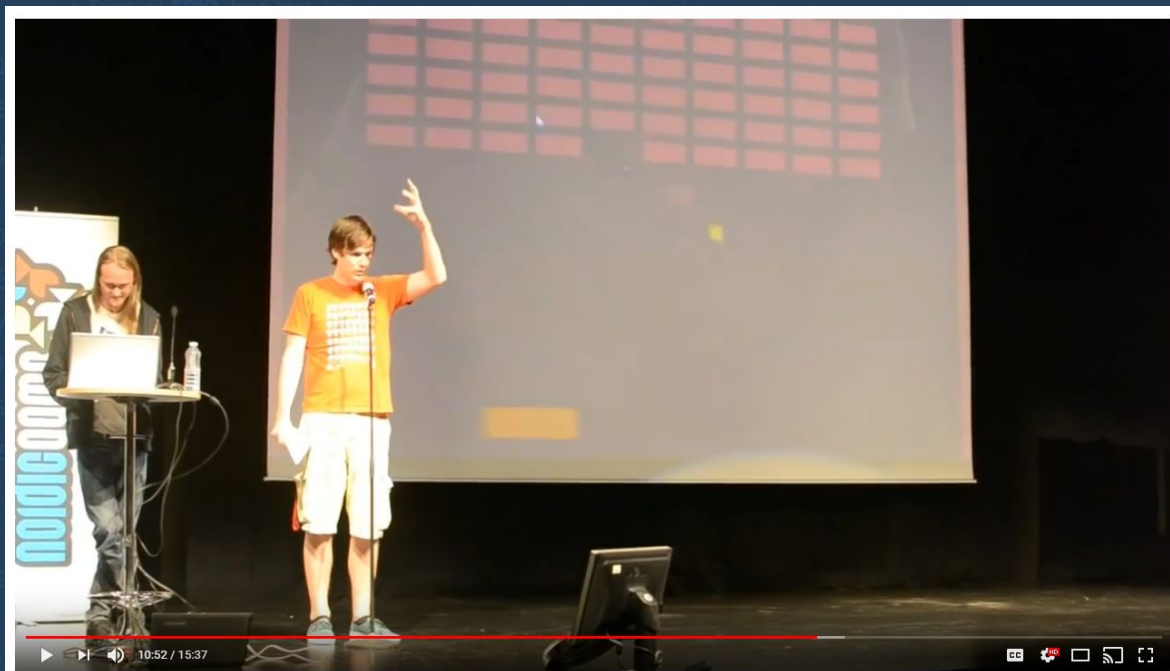
\* @return int the number of the sector (0 if x =

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

# On PC: Hover



# Juiciness



Juice it or lose it - a talk by Martin Jonasson & Petri Purho

<https://www.youtube.com/watch?v=Fy0aCDmgngx>

# Recap

- Recognition vs. recall
- Common imagery
- Affordance
- Mimic physical objects
- Juiciness

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

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

\* Sector will be:



\* @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) {
```

# One last thing...

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



```
function() {
    return [
        [0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0]
    ];
}

function get_sector($x, $y, $z) {
    $x = (int)$x;
    $y = (int)$y;
    $z = (int)$z;

    if ($x < 0 || $x > 7 || $y < 0 || $y > 7 || $z < 0 || $z > 7) {
        return -1;
    }

    return $x + ($y * 8) + ($z * 64);
}
```

# Handledning

[anna@valiant.se](mailto:anna@valiant.se)

@TheAnaka

