

My Project

Alpha

Δημιουργήθηκε από Doxygen 1.8.17

| | | |
|----------|---|----------|
| 1 | Ιεραρχικό Ευρετήριο | 1 |
| 1.1 | Ιεραρχία Κλάσεων | 1 |
| 2 | Συμπαγές Ευρετήριο | 3 |
| 2.1 | Λίστα Κλάσεων | 3 |
| 3 | Τεκμηρίωση Κλάσεων | 5 |
| 3.1 | Τεκμηρίωση Κλάσης <code>Axe</code> | 5 |
| 3.2 | Τεκμηρίωση Κλάσης <code>Coconut</code> | 5 |
| 3.2.1 | Λεπτομερής Περιγραφή | 5 |
| 3.3 | Τεκμηρίωση Κλάσης <code>CSVRow</code> | 6 |
| 3.3.1 | Λεπτομερής Περιγραφή | 6 |
| 3.3.2 | Τεκμηρίωση Συναρτήσεων Μελών | 6 |
| 3.3.2.1 | <code>operator[]()</code> | 6 |
| 3.3.2.2 | <code>readNextRow()</code> | 6 |
| 3.3.2.3 | <code>size()</code> | 7 |
| 3.4 | Τεκμηρίωση Προτύπου Κλάσης <code>DynamicArray< T ></code> | 7 |
| 3.4.1 | Λεπτομερής Περιγραφή | 8 |
| 3.4.2 | Τεκμηρίωση Μελών Ορισμών Τύπων | 8 |
| 3.4.2.1 | <code>iterator</code> | 9 |
| 3.4.2.2 | <code>reference</code> | 9 |
| 3.4.2.3 | <code>size_type</code> | 9 |
| 3.4.3 | Τεκμηρίωση Constructor & Destructor | 9 |
| 3.4.3.1 | <code>DynamicArray()</code> [1/4] | 9 |
| 3.4.3.2 | <code>DynamicArray()</code> [2/4] | 9 |
| 3.4.3.3 | <code>DynamicArray()</code> [3/4] | 10 |
| 3.4.3.4 | <code>DynamicArray()</code> [4/4] | 10 |
| 3.4.3.5 | <code>~DynamicArray()</code> | 10 |
| 3.4.4 | Τεκμηρίωση Συναρτήσεων Μελών | 10 |
| 3.4.4.1 | <code>begin()</code> | 11 |
| 3.4.4.2 | <code>capacity()</code> | 11 |
| 3.4.4.3 | <code>clear()</code> | 11 |
| 3.4.4.4 | <code>emplace_back()</code> | 11 |
| 3.4.4.5 | <code>end()</code> | 12 |
| 3.4.4.6 | <code>erase()</code> | 12 |
| 3.4.4.7 | <code>move_storage()</code> | 12 |
| 3.4.4.8 | <code>operator=()</code> | 14 |
| 3.4.4.9 | <code>operator[]()</code> | 14 |
| 3.4.4.10 | <code>pop_back()</code> | 15 |
| 3.4.4.11 | <code>push_back()</code> | 15 |
| 3.4.4.12 | <code>reallocate()</code> | 15 |
| 3.4.4.13 | <code>size()</code> | 15 |
| 3.4.4.14 | <code>swap()</code> | 16 |

| | |
|---|----|
| 3.5 Τεκμηρίωση Κλάσης Environment | 16 |
| 3.5.1 Λεπτομερής Περιγραφή | 17 |
| 3.5.2 Τεκμηρίωση Constructor & Destructor | 17 |
| 3.5.2.1 Environment() [1/3] | 17 |
| 3.5.2.2 Environment() [2/3] | 17 |
| 3.5.2.3 Environment() [3/3] | 18 |
| 3.5.2.4 ~Environment() | 18 |
| 3.5.3 Τεκμηρίωση Συναρτήσεων Μελών | 18 |
| 3.5.3.1 addItemToGround() [1/2] | 18 |
| 3.5.3.2 addItemToGround() [2/2] | 19 |
| 3.5.3.3 addItemToInv() | 19 |
| 3.5.3.4 generateGrid() | 19 |
| 3.5.3.5 getItemsNearPlayer() | 19 |
| 3.5.3.6 getPlayerCraft() | 20 |
| 3.5.3.7 getPlayerItem() | 20 |
| 3.5.3.8 handleLoadedData() | 20 |
| 3.5.3.9 movePlayer() | 21 |
| 3.5.3.10 removeFromPlayerInv() | 21 |
| 3.5.3.11 removeItemFromGround() | 21 |
| 3.5.4 Τεκμηρίωση Δεδομένων Μελών | 21 |
| 3.5.4.1 lastID | 22 |
| 3.6 Τεκμηρίωση Κλάσης GameScene | 22 |
| 3.7 Τεκμηρίωση Κλάσης Hut | 22 |
| 3.7.1 Λεπτομερής Περιγραφή | 23 |
| 3.8 Τεκμηρίωση Κλάσης Inventory | 23 |
| 3.8.1 Λεπτομερής Περιγραφή | 23 |
| 3.8.2 Τεκμηρίωση Συναρτήσεων Μελών | 23 |
| 3.8.2.1 addItem() | 23 |
| 3.8.2.2 getInventoryItems() | 24 |
| 3.8.2.3 getItemAt() | 24 |
| 3.8.2.4 itemTypeCount() | 24 |
| 3.8.2.5 removeAfterCrafting() | 25 |
| 3.8.2.6 removeAll() | 25 |
| 3.8.2.7 removeItemAt() | 25 |
| 3.9 Τεκμηρίωση Κλάσης IoClass | 25 |
| 3.9.1 Λεπτομερής Περιγραφή | 26 |
| 3.9.2 Τεκμηρίωση Constructor & Destructor | 26 |
| 3.9.2.1 ~IoClass() | 26 |
| 3.9.3 Τεκμηρίωση Συναρτήσεων Μελών | 26 |
| 3.9.3.1 getInput() | 27 |
| 3.9.3.2 getMaxX() | 27 |
| 3.9.3.3 getMaxY() | 27 |

| | |
|---|----|
| 3.9.3.4 loadFromFile() | 27 |
| 3.9.3.5 printEnvironment() | 28 |
| 3.9.3.6 printPlayerStats() | 28 |
| 3.9.3.7 printToCoordsAnimated() | 28 |
| 3.9.3.8 readString() | 29 |
| 3.9.3.9 saveToFile() | 29 |
| 3.9.3.10 showMenu() | 29 |
| 3.10 Τεκμηρίωση Κλάσης Item | 30 |
| 3.10.1 Λεπτομερής Περιγραφή | 30 |
| 3.10.2 Τεκμηρίωση Συναρτήσεων Μελών | 31 |
| 3.10.2.1 operator+() | 31 |
| 3.11 Τεκμηρίωση Κλάσης Leafs | 31 |
| 3.11.1 Λεπτομερής Περιγραφή | 32 |
| 3.12 Τεκμηρίωση Κλάσης LightedTorch | 32 |
| 3.12.1 Λεπτομερής Περιγραφή | 32 |
| 3.13 Τεκμηρίωση Κλάσης Lighter | 32 |
| 3.13.1 Λεπτομερής Περιγραφή | 33 |
| 3.14 Τεκμηρίωση Κλάσης OpenCoconut | 33 |
| 3.14.1 Λεπτομερής Περιγραφή | 33 |
| 3.15 Τεκμηρίωση Κλάσης Player | 33 |
| 3.15.1 Λεπτομερής Περιγραφή | 34 |
| 3.15.2 Τεκμηρίωση Constructor & Destructor | 34 |
| 3.15.2.1 Player() | 34 |
| 3.15.3 Τεκμηρίωση Συναρτήσεων Μελών | 35 |
| 3.15.3.1 addToInventory() | 35 |
| 3.15.3.2 getCrafted() | 35 |
| 3.15.3.3 getHunger() | 35 |
| 3.15.3.4 getInventory() | 36 |
| 3.15.3.5 getItemAt() | 36 |
| 3.15.3.6 getItems() | 36 |
| 3.15.3.7 getName() | 36 |
| 3.15.3.8 getPosition() | 37 |
| 3.15.3.9 moveToCoordinates() [1/2] | 37 |
| 3.15.3.10 moveToCoordinates() [2/2] | 37 |
| 3.15.3.11 removeAfterCrafting() | 37 |
| 3.15.3.12 removeAllFromPlayer() | 38 |
| 3.15.3.13 removeFromInventory() | 38 |
| 3.15.3.14 showInventory() | 38 |
| 3.16 Τεκμηρίωση Κλάσης Rock | 39 |
| 3.16.1 Λεπτομερής Περιγραφή | 39 |
| 3.17 Τεκμηρίωση Προτύπου Κλάσης Vector2D< T > | 39 |
| 3.17.1 Λεπτομερής Περιγραφή | 40 |

| | |
|--|-----------|
| 3.17.2 Τεκμηρίωση Constructor & Destructor | 40 |
| 3.17.2.1 Vector2D() [1/3] | 41 |
| 3.17.2.2 Vector2D() [2/3] | 41 |
| 3.17.2.3 Vector2D() [3/3] | 41 |
| 3.17.3 Τεκμηρίωση Συναρτήσεων Μελών | 41 |
| 3.17.3.1 length() | 41 |
| 3.17.3.2 normalize() | 42 |
| 3.17.3.3 operator*() | 42 |
| 3.17.3.4 operator*=() | 42 |
| 3.17.3.5 operator+() [1/2] | 43 |
| 3.17.3.6 operator+() [2/2] | 43 |
| 3.17.3.7 operator+=() [1/2] | 44 |
| 3.17.3.8 operator+=() [2/2] | 44 |
| 3.17.3.9 operator-() [1/2] | 44 |
| 3.17.3.10 operator-() [2/2] | 45 |
| 3.17.3.11 operator-=() [1/2] | 45 |
| 3.17.3.12 operator-=() [2/2] | 45 |
| 3.17.3.13 operator/() | 46 |
| 3.17.3.14 operator/=() | 46 |
| 3.17.3.15 operator=() | 47 |
| 3.17.3.16 rotate() | 47 |
| 3.17.3.17 set() | 47 |
| 3.18 Τεκμηρίωση Κλάσης WoodStick | 48 |
| 3.18.1 Λεπτομερής Περιγραφή | 48 |
| Index | 49 |

Chapter 1

Ιεραρχικό Ευρετήριο

1.1 Ιεραρχία Κλάσεων

Αυτή η λίστα κληρονομικότητας είναι μερικώς ταξινομημένη αλφαβητικά:

| | |
|--------------------------------|----|
| CSVRow | 6 |
| DynamicArray< T > | 7 |
| DynamicArray< Item > | 7 |
| Environment | 16 |
| GameScene | 22 |
| Inventory | 23 |
| IoClass | 25 |
| Item | 30 |
| Axe | 5 |
| Coconut | 5 |
| Hut | 22 |
| Leafs | 31 |
| LightedTorch | 32 |
| Lighter | 32 |
| OpenCoconut | 33 |
| Rock | 39 |
| WoodStick | 48 |
| Player | 33 |
| Vector2D< T > | 39 |
| Vector2D< int > | 39 |

Chapter 2

Συμπαγές Ευρετήριο

2.1 Λίστα Κλάσεων

Ακολουθούν οι κλάσεις, οι δομές, οι ενώσεις και οι διασυνδέσεις με σύντομες περιγραφές:

| | | |
|--------------------------------|--|----|
| Axe | Κλάση αντικειμένου Axe Χρησιμοποιείται για τα συμβολίσει Τσεκούρι | 5 |
| Coconut | Κλάση αντικειμένου Coconut Χρησιμοποιείται για τα συμβολίσει Καρύδα | 5 |
| CSVRow | Κλάση CSVRow χρησιμοποιείται για την προσπέλαση και ανάγνωση δεδομένων απο csv αρχεία | 6 |
| DynamicArray< T > | Κλάση δυναμικού πίνακα που χρησιμοποιεί templates . Για την υλοποίηση συμβουλευθήκα το βιβλίο του Stroustrup "Προγραμματισμός με τη C++" | 7 |
| Environment | Κλάση περιβάλλοντος παιχνιδιού | 16 |
| GameScene | | 22 |
| Hut | Κλάση αντικειμένου Hut Χρησιμοποιείται για τα συμβολίσει καλύβα(θα προστεθεί στο μέλλον) | 22 |
| Inventory | Κλάση που περιέχει τα αντικείμενα του χρήστη | 23 |
| IoClass | Κλάση διαχείρισης εισοδου-εξόδου καθώς και διαχείρισης οθόνης | 25 |
| Item | Κλάση που χρησιμοποιείται για το κάθε αντικείμενο του παιχνιδιού | 30 |
| Leafs | Κλάση αντικειμένου Leafs Χρησιμοποιείται για τα συμβολίσει Φύλλο | 31 |
| LightedTorch | Κλάση αντικειμένου LightedTorch Χρησιμοποιείται για τα συμβολίσει αναμμένο πυρσό | 32 |
| Lighter | Κλάση αντικειμένου Lighter Χρησιμοποιείται για τα συμβολίσει αναπτήρα | 32 |
| OpenCoconut | Κλάση αντικειμένου OpenCoconut Χρησιμοποιείται για τα συμβολίσει ανοικτή καρύδα | 33 |

| | | |
|-------------------------------------|--|----|
| Player | Κλάση για τον παίκτη του παιχνιδιού | 33 |
| Rock | Κλάση αντικειμένου Rock Χρησιμοποιείται για τα συμβολίζει πέτρα | 39 |
| Vector2D< T > | Κλάση για διάνυσμα δυσδιάστατου χώρου που είχα σχεδιάσει στο παρελθόν στα πλαίσια εξάσκησης για τα <code>templates</code> | 39 |
| WoodStick | Κλάση αντικειμένου WoodStick Χρησιμοποιείται για τα συμβολίζει ξύλο | 48 |

Chapter 3

Τεκμηρίωση Κλάσεων

3.1 Τεκμηρίωση Κλάσης **Axe**

Κλάση αντικειμένου **Axe**

Χρησιμοποιείται για τα συμβολίζει Τσεκούρι

```
#include <Axe.h>
```

Διάγραμμα κληρονομικότητας για την **Axe**:

3.2 Τεκμηρίωση Κλάσης **Coconut**

Κλάση αντικειμένου **Coconut**

Χρησιμοποιείται για τα συμβολίζει Καρύδα

```
#include <Coconut.h>
```

Διάγραμμα κληρονομικότητας για την **Coconut**:

Διάγραμμα Συνεργασίας για την κλάση **Coconut**:

Δημόσιες Μέθοδοι

- **Coconut** (string name, string id, **Vector2D**< int >pos, bool isOnFloor=true)
- **Coconut** (const **Coconut** ©)

3.2.1 Λεπτομερής Περιγραφή

Κλάση αντικειμένου **Coconut**

Χρησιμοποιείται για τα συμβολίζει Καρύδα

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- Items/Coconut/Coconut.h
- Items/Coconut/Coconut.cpp

3.3 Τεκμηρίωση Κλάσης **CSVRow**

Κλάση **CSVRow** χρησιμοποιείται για την προσπέλαση και ανάγνωση δεδομένων απο csv αρχεία

```
#include <CSVRow.h>
```

Δημόσιες Μέθοδοι

- `std::string const & operator[] (std::size_t index) const`
Επιστρέφει *string* που βρίσκεται στο συγκεκριμένο *index*.
- `std::size_t size () const`
Επιστρέφει το μέγεθος του *vector* που περιέχει τις γραμμές
- `void readNextRow (std::istream &str)`
Διαβάζει την επόμενη γραμμή απο *istream str*.

3.3.1 Λεπτομερής Περιγραφή

Κλάση **CSVRow** χρησιμοποιείται για την προσπέλαση και ανάγνωση δεδομένων απο csv αρχεία

3.3.2 Τεκμηρίωση Συναρτήσεων Μελών

3.3.2.1 `operator[]()`

```
std::string const& CSVRow::operator[] (
    std::size_t index ) const [inline]
```

Επιστρέφει *string* που βρίσκεται στο συγκεκριμένο *index*.

Παράμετροι

| | |
|--------------|--------------------------|
| <i>index</i> | Η θέση του <i>string</i> |
|--------------|--------------------------|

Επιστρέφει

`std::string const&`

3.3.2.2 `readNextRow()`

```
void CSVRow::readNextRow (
    std::istream & str ) [inline]
```

Διαβάζει την επόμενη γραμμή απο *istream str*.

Παράμετροι

| | |
|------------|--|
| <i>str</i> | Το input stream απο το οποίο θα διαβάσει |
|------------|--|

3.3.2.3 size()

```
std::size_t CSVRow::size ( ) const [inline]
```

Επιστρέφει το μέγεθος του **vector** που περιέχει τις γραμμές

Επιστρέφει

`std::size_t`

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από το ακόλουθο αρχείο:

- CSVRow/CSVRow.h

3.4 Τεκμηρίωση Προτύπου Κλάσης **DynamicArray< T >**

Κλάση δυναμικού πίνακα που χρησιμοποιεί **templates**. Για την υλοποίηση συμβουλευθήκα το βιβλίο του Stroustrup "Προγραμματισμός με τη C++".

```
#include <DynamicArray.h>
```

Δημόσιοι Τυποι

- using **size_type** = `size_t`
- using **iterator** = `T *`
- using **reference** = `T &`

Δημόσιες Μέθοδοι

- void **swap** (**DynamicArray**< T > &rhs)
Χρησιμοποιείται για να κάνει **swap** τα δεδομένα του αντικειμένου που το καλεί με τα δεδομένα της παραμέτρου.
- **DynamicArray** () noexcept
Δημιουργία ενός νέου αντικειμένου **DynamicArray**. Εάν κληθεί αυτός ο **constructor** τότε δεσμεύω χώρο χρησιμοποιώντας το *m_capacity*. Δηλαδή δεσμεύει χώρο ακόμα και εάν ο δυναμικός πίνακας είναι κενός.
- **DynamicArray** (std::initializer_list< T > init)
Δημιουργία ενός νέου αντικειμένου **DynamicArray**. Παίρνει ως όρισμα ένα *initializer_list* και αυτό για να γίνεται *initialize dynamicVector<int> a({1,3,2});*.
- **DynamicArray**< T > & **operator=** (const **DynamicArray**< T > &origin)
Υπερφόρτωση του τελεστή =.
- **DynamicArray** (const **DynamicArray**< T > &origin)
Δημιουργία ενός νέου αντικειμένου **DynamicArray** χρησιμοποιώντας *copy constructor*.

- **DynamicArray** (**DynamicArray**< T > &&origin)
Δημιουργία ενός νέου αντικειμένου **DynamicArray** χρησιμοποιώντας *copy constructor*.
- void **clear** () noexcept
Χρησιμοποιείται για να αδειάσει ο δυναμικός πίνακας. Δεν είναι ο *destructor*.
- template<typename... Args>
DynamicArray< T >::reference **emplace_back** (Args &&... args)
Χρησιμοποιείται από την *push_back* κυρίως για να προσθέτει αντικείμενα στο τέλος του πίνακα.
- void **push_back** (const T &val)
Προσθέτει ένα αντικείμενο στο τέλος του πίνακα
- **DynamicArray**< T >::iterator **erase** (**DynamicArray**< T >::iterator iter)
Διαγράφω ένα συγκεκριμένο αντικείμενο χρησιμοποιώντας *iterator(T*)*
- void **reallocate** ()
Χρησιμοποιείται από την ίδια την κλάση για να δεσμεύσει περισσότερο χώρο.
- void **move_storage** (T *dest, T *from, **size_type** n)
Χρησιμοποιείται για να "μεταφέρει" δεδομένα χρησιμοποιώντας την *move*.
- **DynamicArray**< T >::iterator **begin** () noexcept
Επιστρέφει *iterator* του πρώτου αντικειμένου
- **DynamicArray**< T >::iterator **end** () const noexcept
Επιστρέφει *iterator* του τελευταίου αντικειμένου
- **size_t** **size** ()
Getter μεγέθους(πραγματικού)
- **size_t** **capacity** ()
Getter μεγέθους(*extra*)
- T & **operator[]** (**size_t** N)
Υπερφόρτωση του τελεστή [].
- T **pop_back** ()
Αντίστοιχη της *pop_back* του STL Vector.
- ~**DynamicArray** ()
destructor

Φίλοι

- void **swap** (**DynamicArray**< T > &lhs, **DynamicArray**< T > &rhs)

3.4.1 Λεπτομερής Περιγραφή

```
template<typename T>
class DynamicArray< T >
```

Κλάση δυναμικού πίνακα που χρησιμοποιεί **templates**. Για την υλοποίηση συμβουλευθήκα το βιβλίο του Stroustrup "Προγραμματισμός με τη C++".

Μια πολύ βασική υλοποίηση δυναμικού πίνακα που έχει αρκετές ομοιότητες με το διάνυσμα της STL.

3.4.2 Τεκμηρίωση Μελών Ορισμών Τύπων

3.4.2.1 iterator

```
template<typename T >
using DynamicArray< T >::iterator = T *
```

Χρησιμοποιείται σαν ψευδώνυμο για το T*

3.4.2.2 reference

```
template<typename T >
using DynamicArray< T >::reference = T &
```

Χρησιμοποιείται σαν ψευδώνυμο για το reference

3.4.2.3 size_type

```
template<typename T >
using DynamicArray< T >::size_type = size_t
```

Χρησιμοποιείται σαν ψευδώνυμο για το size_t

3.4.3 Τεκμηρίωση Constructor & Destructor**3.4.3.1 DynamicArray() [1/4]**

```
template<typename T >
DynamicArray< T >::DynamicArray ( ) [inline], [noexcept]
```

Δημιουργία ενός νέου αντικειμένου DynamicArray. Εάν κληθεί αυτός ο constructor τότε δεσμεύω χώρο χρησιμοποιώντας το m_capacity. Δηλαδή δεσμεύει χώρο ακόμα και εάν ο δυναμικός πίνακας είναι κενός.

3.4.3.2 DynamicArray() [2/4]

```
template<typename T >
DynamicArray< T >::DynamicArray (
    std::initializer_list< T > init ) [inline]
```

Δημιουργία ενός νέου αντικειμένου DynamicArray. Παίρνει ως όρισμα ένα initializer_list και αυτό για να γίνεται initialize dynamicVector<int> a({1,3,2});.

Παράμετροι

| | |
|-------------|---|
| <i>init</i> | Η initializer_list με την οποία θα γίνει η αρχικοποίηση |
|-------------|---|

3.4.3.3 `DynamicArray()` [3/4]

```
template<typename T >
DynamicArray< T >::DynamicArray (
    const DynamicArray< T > & origin ) [inline]
```

Δημιουργία ενός νέου αντικειμένου `DynamicArray` χρησιμοποιώντας copy constructor.

Παράμετροι

| | |
|---------------|---|
| <i>origin</i> | Reference στο αντικείμενο απο το οποίο θα αντιγραφούν τα δεδομένα |
|---------------|---|

3.4.3.4 `DynamicArray()` [4/4]

```
template<typename T >
DynamicArray< T >::DynamicArray (
    DynamicArray< T > && origin ) [inline]
```

Δημιουργία ενός νέου αντικειμένου `DynamicArray` χρησιμοποιώντας copy constructor.

Παράμετροι

| | |
|---------------|--|
| <i>origin</i> | Rvalue reference στο αντικείμενο απο το οποίο θα αντιγραφούν τα δεδομένα |
|---------------|--|

3.4.3.5 `~DynamicArray()`

```
template<typename T >
DynamicArray< T >::~~DynamicArray ( ) [inline]
```

destructor

3.4.4 Τεκμηρίωση Συναρτήσεων Μελών

3.4.4.1 begin()

```
template<typename T >
DynamicArray<T>::iterator DynamicArray< T >::begin ( ) [inline], [noexcept]
```

Επιστρέφει iterator του πρώτου αντικειμένου

Επιστρέφει

DynamicArray<T>::iterator iterator που δείχνει στο πρώτο αντικείμενο

3.4.4.2 capacity()

```
template<typename T >
size_t DynamicArray< T >::capacity ( ) [inline]
```

Getter μεγέθους(extra)

Επιστρέφει

size_t Το extra μέγεθος.

3.4.4.3 clear()

```
template<typename T >
void DynamicArray< T >::clear ( ) [inline], [noexcept]
```

Χρησιμοποιείται για να αδειάσει ο δυναμικός πίνακας. Δεν είναι ο destructor.

3.4.4.4 emplace_back()

```
template<typename T >
template<typename... Args>
DynamicArray<T>::reference DynamicArray< T >::emplace_back (
    Args &&... args ) [inline]
```

Χρησιμοποιείται απο την push_back κυρίως για να προσθέτει αντικείμενα στο τέλος του πίνακα.

Παράμετροι Προτύπου

| | |
|-------------|---|
| Args | Ο τύπος των αντικειμένων που θα προστεθεί |
|-------------|---|

Παράμετροι

| | |
|-------------|---|
| <i>args</i> | Αναφορά στα αντικείμενα που θα προστεθούν |
|-------------|---|

Επιστρέφει

[DynamicArray<T>::reference](#) Επιστρέφει αναφορά στον πίνακα

3.4.4.5 end()

```
template<typename T >
DynamicArray<T>::iterator DynamicArray< T >::end ( ) const [inline], [noexcept]
```

Επιστρέφει *iterator* του τελευταίου αντικειμένου

Επιστρέφει

[DynamicArray<T>::iterator](#) *iterator* που δείχνει στο τελευταίο αντικείμενο

3.4.4.6 erase()

```
template<typename T >
DynamicArray<T>::iterator DynamicArray< T >::erase (
    DynamicArray< T >::iterator iter ) [inline]
```

Διαγράφω ένα συγκεκριμένο αντικείμενο χρησιμοποιώντας *iterator(T*)*

Παράμετροι

| | |
|-------------|---|
| <i>iter</i> | Ο <i>iterator</i> που θα χρησιμοποιηθεί για να διαγραφεί το αντικείμενο απο τον πίνακα. |
|-------------|---|

Επιστρέφει

[DynamicArray<T>::iterator](#) Επιστρέφει *iterator* όπως ακριβώς γίνεται και στον *vector* της STL

3.4.4.7 move_storage()

```
template<typename T >
void DynamicArray< T >::move_storage (
    T * dest,
```

```
T * from,  
size_type n ) [inline]
```

Χρησιμοποιείται για να "μεταφέρει" δεδομένα χρησιμοποιώντας την `move`.

Παράμετροι

| | |
|-------------|----------------------|
| <i>dest</i> | Διεύθυνση προορισμού |
| <i>from</i> | Διεύθυνση πηγής |
| <i>n</i> | Πλήθος/Μέγεθος |

3.4.4.8 operator=()

```
template<typename T >
DynamicArray<T>& DynamicArray< T >::operator= (
    const DynamicArray< T > & origin ) [inline]
```

Υπερφόρτωση του τελεστή =.

Παράμετροι

| | |
|---------------|---|
| <i>origin</i> | Το αντικείμενο με το οποίο θα γίνει η υπερφόρτωση |
|---------------|---|

Επιστρέφει

DynamicArray<T>&

3.4.4.9 operator[]()

```
template<typename T >
T& DynamicArray< T >::operator[] (
    size_t N ) [inline]
```

Υπερφόρτωση του τελεστή [].

Παράμετροι

| | |
|----------|---|
| <i>N</i> | Ο αριθμός/index του αντικειμένου που θέλουμε να πάρουμε |
|----------|---|

Επιστρέφει

T& Το αντικείμενο που βρίσκεται στο συγκεκριμένο index

3.4.4.10 pop_back()

```
template<typename T >
T DynamicArray< T >::pop_back ( ) [inline]
```

Αντίστοιχη της pop_back του STL Vector.

Επιστρέφει

T Το αντικείμενο που έγινε pop

3.4.4.11 push_back()

```
template<typename T >
void DynamicArray< T >::push_back (
    const T & val ) [inline]
```

Προσθέτει ένα αντικείμενο στο τέλος του πίνακα

Παράμετροι

| | |
|------------|--|
| <i>val</i> | Το αντικείμενο που θέλουμε να προστεθεί. |
|------------|--|

3.4.4.12 reallocate()

```
template<typename T >
void DynamicArray< T >::reallocate ( ) [inline]
```

Χρησιμοποιείται απο την ίδια την κλάση για να δεσμεύσει περισσότερο χώρο.

3.4.4.13 size()

```
template<typename T >
size_t DynamicArray< T >::size ( ) [inline]
```

Getter μεγέθους(πραγματικού)

Επιστρέφει

size_t Το πραγματικό μέγεθος

3.4.4.14 swap()

```
template<typename T >
void DynamicArray< T >::swap (
    DynamicArray< T > & rhs ) [inline]
```

Χρησιμοποιείται για να κάνει **swap** τα δεδομένα του αντικειμένου που το καλεί με τα δεδομένα της παραμέτρου.

Παράμετροι

| | |
|------------|---|
| <i>rhs</i> | Ο δυναμικός πίνακας με τον οποίο θα κάνει swap |
|------------|---|

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από το ακόλουθο αρχείο:

- DynamicArray/DynamicArray.h

3.5 Τεκμηρίωση Κλάσης **Environment**

Κλάση περιβάλλοντος παιχνιδιού.

```
#include <Environment.h>
```

Δημόσιες Μέθοδοι

- **Environment** ()
Δημιουργεί ένα αντικείμενο *Environment*.
- **Environment** (const *Environment* &env)
Δημιουργεί ένα αντικείμενο *Environment* με την χρήση ενός άλλου αντικειμένου *Environment*.
- **Environment** (*Player* &player, int X, int Y, int lastid)
Δημιουργεί ένα αντικείμενο *Environment*.
- **~Environment** ()
Destructor.
- void **movePlayer** (int x, int y)
Μετακινεί τον παίκτη στις συγκεκριμένες συντεταγμένες
- void **addItemToInv** (*Item* &itemPtr)
Προσθέτει το αντικείμενο στον πίνακα με τα αντικείμενα του χρήστη. Σημαντική σημείωση : Το αφαιρεί από το πάτωμα πρώτου το μετακινήσει.
- void **addItemToGround** (*Item* &item)
Προσθέτει το αντικείμενο στον πίνακα με τα αντικείμενα που βρίσκονται στο πάτωμα
- void **addItemToGround** (vector< *Item* > items)
Προσθέτει τα αντικείμενα που βρίσκονται στον *vector* στον πίνακα με τα αντικείμενα του πατώματος
- void **removeFromPlayerInv** (*Item* &itemPtr)
Διαγράφει το αντικείμενο από τον πίνακα με τα αντικείμενα του χρήστη.
- void **removeItemFromGround** (*Item* &itemPtr)
Διαγράφει το αντικείμενο από τον πίνακα με τα αντικείμενα που βρίσκονται στο πάτωμα
- void **generateGrid** (int numberOfEntities)
Δημιουργεί τυχαία αντικείμενα πλήθους ίσου με την παράμετρο *numberOfEntities*.
- vector< *Item* > **getGroundItems** ()

- `DynamicArray< Item > getPlayerItems ()`
- `Player & getPlayer ()`
- `Item & getItemAt (int X, int Y)`
- `char ** getGrid ()`
- `int getX ()`
- `int getY ()`
- `Item & getPlayerItem (int index)`
Επιστρέφει αναφορά σε αντικείμενο που βρίσκεται σε συγκεκριμένο *index*.
- `vector< Item > getItemsNearPlayer ()`
Επιστρέφει *vector* με τα αντικείμενα που βρίσκονται 1 θέση πάνω/κάτω/δεξιά/αριστερά και διαγώνια από τον χρήστη
- `DynamicArray< Item > getPlayerCraft ()`
Επιστρέφει δυναμικό πίνακα που περιέχει αντικείμενα που μπορεί να δημιουργήσει ο χρήστης
- `void handleLoadedData (vector< Item > ldata, bool isStartup)`
Μέθοδος που διαχειρίζεται τα δεδομένα που φορτώθηκαν από αρχείο

Στατικά Δημόσια Χαρακτηριστικά

- `static int lastID = 0`

3.5.1 Λεπτομερής Περιγραφή

Κλάση περιβάλλοντος παιχνιδιού.

3.5.2 Τεκμηρίωση **Constructor & Destructor**

3.5.2.1 **Environment()** [1/3]

```
Environment::Environment ( )
```

Δημιουργεί ένα αντικείμενο **Environment**.

3.5.2.2 **Environment()** [2/3]

```
Environment::Environment (
    const Environment & env )
```

Δημιουργεί ένα αντικείμενο **Environment** με την χρήση ενός άλλου αντικειμένου **Environment**.

Παράμετροι

| | |
|------------|--|
| <i>env</i> | Το αντικείμενο που θα χρησιμοποιηθεί στον copy constructor |
|------------|--|

3.5.2.3 Environment() [3/3]

```
Environment::Environment (
    Player & player,
    int X,
    int Y,
    int lastid )
```

Δημιουργεί ένα αντικείμενο **Environment**.

Παράμετροι

| | |
|---------------|--|
| <i>player</i> | Αναφορά σε αντικείμενο τύπου player |
| <i>X</i> | Το μήκος του Grid |
| <i>Y</i> | Το πλάτος του Grid |
| <i>lastid</i> | Το ID τελευταίου αντικειμένου |

3.5.2.4 ~Environment()

```
Environment::~~Environment ( )
```

Destructor.

3.5.3 Τεκμηρίωση Συναρτήσεων Μελών

3.5.3.1 addItemToGround() [1/2]

```
void Environment::addItemToGround (
    Item & item )
```

Προσθέτει το αντικείμενο στον πίνακα με τα αντικείμενα που βρίσκονται στο πάτωμα

Παράμετροι

| | |
|-------------|-------------------------|
| <i>item</i> | Αναφορά στο αντικείμενο |
|-------------|-------------------------|

3.5.3.2 addItemToGround() [2/2]

```
void Environment::addItemToGround (
    vector< Item > items )
```

Προσθέτει τα αντικείμενα που βρίσκονται στον **vector** στον πίνακα με τα αντικείμενα του πατώματος

Παράμετροι

| | |
|-------------|-------------------------|
| <i>item</i> | Αναφορά στο αντικείμενο |
|-------------|-------------------------|

3.5.3.3 addItemToInv()

```
void Environment::addItemToInv (
    Item & itemPtr )
```

Προσθέτει το αντικείμενο στον πίνακα με τα αντικείμενα του χρήστη. Σημαντική σημείωση : Το αφαιρεί από το πάτωμα πρώτου το μετακινήσει.

Παράμετροι

| | |
|----------------|-------------------------|
| <i>itemPtr</i> | Αναφορά στο αντικείμενο |
|----------------|-------------------------|

3.5.3.4 generateGrid()

```
void Environment::generateGrid (
    int numberOfEntities )
```

Δημιουργεί τυχαία αντικείμενα πλήθους ίσου με την παράμετρο **numberOfEntities**.

Παράμετροι

| | |
|-------------------------|---|
| <i>numberOfEntities</i> | Το πλήθος των τυχαίων αντικειμένων που επιθυμεί ο χρήστης να δημιουργήσει |
|-------------------------|---|

3.5.3.5 getItemsNearPlayer()

```
vector< Item > Environment::getItemsNearPlayer ( )
```

Επιστρέφει **vector** με τα αντικείμενα που βρίσκονται 1 θέση πάνω/κάτω/δεξιά/αριστερά και διαγώνια από τον χρήστη

Επιστρέφει

`vector<Item>`

3.5.3.6 getPlayerCraft()

```
DynamicArray< Item > Environment::getPlayerCraft ( )
```

Επιστρέφει δυναμικο πίνακα που περιέχει αντικείμενα που μπορεί να δημιουργήσει ο χρήστης

Επιστρέφει

`DynamicArray<Item>` Δυναμικός πίνακας που έχει το περιεχόμενο

3.5.3.7 getPlayerItem()

```
Item & Environment::getPlayerItem (
    int index )
```

Επιστρέφει αναφορά σε αντικείμενο που βρίσκεται σε συγκεκριμένο index.

Παράμετροι

| | |
|--------------|---|
| <i>index</i> | Το index του αντικειμένου στον πίνακα αντικειμένων. |
|--------------|---|

Επιστρέφει

`Item&` Αναφορά στο αντικείμενο.

3.5.3.8 handleLoadedData()

```
void Environment::handleLoadedData (
    vector< Item > ldata,
    bool isStartup )
```

Μέθοδος που διαχειρίζεται τα δεδομένα που φορτώθηκαν απο αρχείο

Παράμετροι

| | |
|------------------|--|
| <i>ldata</i> | vector που περιέχει τα δεδομένα που διαβάστηκαν |
| <i>isStartup</i> | Flag που είναι 1 εαν πρόκειται για startup αλλιώς 0. |

3.5.3.9 movePlayer()

```
void Environment::movePlayer (
    int x,
    int y )
```

Μετακινεί τον παίκτη στις συγκεκριμένες συντεταγμένες

Παράμετροι

| | |
|----------|--|
| <i>x</i> | |
| <i>y</i> | |

3.5.3.10 removeFromPlayerInv()

```
void Environment::removeFromPlayerInv (
    Item & itemptr )
```

Διαγράφει το αντικείμενο απο τον πίνακα με τα αντικείμενα του χρήστη.

Παράμετροι

| | |
|----------------|--|
| <i>itemptr</i> | Το αντικείμενο που θέλουμε να διαγραφεί. |
|----------------|--|

3.5.3.11 removeItemFromGround()

```
void Environment::removeItemFromGround (
    Item & itemptr )
```

Διαγράφει το αντικείμενο απο τον πίνακα με τα αντικείμενα που βρίσκονται στο πάτωμα

Παράμετροι

| | |
|----------------|--|
| <i>itemptr</i> | |
|----------------|--|

3.5.4 Τεκμηρίωση Δεδομένων Μελών

3.5.4.1 lastID

```
int Environment::lastID = 0 [static]
```

Χρησιμοποιείται για την "διευθυνσιοδότηση" των αντικειμένων ώστε να μην υπάρχουν αντικείμενα με το ίδιο ID

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- Environment/Environment.h
- Environment/Environment.cpp

3.6 Τεκμηρίωση Κλάσης **GameScene**

Δημόσιες Μέθοδοι

- **GameScene** (IoClass *ioManager, Environment *env)
- **GameScene** (const GameScene ©)
- GameState **getState** ()
- void **checkHunger** (chrono::minutes::rep &, chrono::_V2::system_clock::time_point &)
- void **setState** (GameState state)
- string **startupScreen** ()
- void **Play** ()
- void **parseSelection** (int c)
- void **handleMainMenu** (int menuSelection)
- void **handleInventoryMenu** (int menuSelection)
- void **handleCraftingMenu** (int menuSelection)
- void **handleEndingMenu** (int menuSelection)

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- GameScene/GameScene.h
- GameScene/GameScene.cpp

3.7 Τεκμηρίωση Κλάσης **Hut**

Κλάση αντικειμένου **Hut**

Χρησιμοποιείται για τα συμβολίζει καλύβα(θα προστεθεί στο μέλλον)

```
#include <Hut.h>
```

Διάγραμμα κληρονομικότητας για την Hut:

Διάγραμμα Συνεργασίας για την κλάση Hut:

Δημόσιες Μέθοδοι

- **Hut** (string name, string id, Vector2D< int > pos, bool isOnFloor=true)
- **Hut** (const Hut ©)

3.7.1 Λεπτομερής Περιγραφή

Κλάση αντικειμένου **Hut**

Χρησιμοποιείται για τα συμβολίσει καλύβα(θα προστεθεί στο μέλλον)

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- Items/Hut/Hut.h
- Items/Hut/Hut.cpp

3.8 Τεκμηρίωση Κλάσης **Inventory**

Κλάση που περιέχει τα αντικείμενα του χρήστη.

```
#include <Inventory.h>
```

Δημόσιες Μέθοδοι

- **Inventory** (const **Inventory** ©)
- void **addItem** (**Item** &item)
Προσθέτει ένα αντικείμενο στο τέλος του δυναμικού πίνακα που περιέχει τα αντικείμενα.
- **DynamicArray**< **Item** > **getInventoryItems** ()
Getter για τον δυναμικό πίνακα αντικειμένων
- **Item** & **getItemAt** (int index)
Getter για ένα συγκεκριμένο αντικείμενο
- void **removeItemAt** (int index)
Διαγράφει αντικείμενο στη συγκεκριμένη θέση που προσδιορίζεται από την παράμετρο *index*.
- **DynamicArray**< int > **itemTypeCount** ()
Επιστρέφει δυναμικό πίνακα που περιέχει το πλήθος των αντικειμένων του κάθε τύπου
Για παράδειγμα εάν έχουμε 5 αντικείμενα τύπου *woodstick* τότε στην αντίστοιχη θέση του τύπου *woodstick* θα περιέχει 5.
- size_t **getSize** ()
- **Inventory** & **operator=** (**Inventory** &inv)
- void **removeAll** ()
Διαγράφει όλα τα αντικείμενα που βρίσκονται στον δυναμικό πίνακα
- **DynamicArray**< **Item** > **removeAfterCrafting** (itemType itemTypeCraftedType)
Επιστρέφει δυναμικό πίνακα που περιέχει τα πρωτογενή αντικείμενα που διαγράφηκαν για να κάνει *craft* ο χρήστης.
Εάν ο χρήστης για παράδειγμα δημιουργήσει ένα *Axe* από *Rock* και *woodstick* τότε θα προσθέσει τα 2 αντικείμενα αυτά στον δυναμικό πίνακα και μετά θα τα διαγράψει. Τέλος θα επιστεφει τον πίνακα.

3.8.1 Λεπτομερής Περιγραφή

Κλάση που περιέχει τα αντικείμενα του χρήστη.

3.8.2 Τεκμηρίωση Συναρτήσεων Μελών

3.8.2.1 addItem()

```
void Inventory::addItem (  
    Item & item )
```

Προσθέτει ένα αντικείμενο στο τέλος του δυναμικού πίνακα που περιέχει τα αντικείμενα.

Παράμετροι

| | |
|-------------|--|
| <i>item</i> | |
|-------------|--|

3.8.2.2 `getInventoryItems()`

```
DynamicArray< Item > Inventory::getInventoryItems ( )
```

Getter για τον δυναμικό πίνακα αντικειμένων

Επιστρέφει

`DynamicArray<Item>`

3.8.2.3 `getItemAt()`

```
Item & Inventory::getItemAt (
    int index )
```

Getter για ένα συγκεκριμένο αντικείμενο

Παράμετροι

| | |
|--------------|---------------------------|
| <i>index</i> | το index του αντικειμένου |
|--------------|---------------------------|

Επιστρέφει

`Item`& αναφορά στο αντικείμενο που βρίσκεται στη συγκεκριμένη θέση

3.8.2.4 `itemTypeCount()`

```
DynamicArray< int > Inventory::itemTypeCount ( )
```

Επιστρέφει δυναμικό πίνακα που περιέχει το πλήθος των αντικειμένων του κάθε τύπου

Για παράδειγμα εάν έχουμε 5 αντικείμενα τύπου `woodstick` τότε στην αντίστοιχη θέση του τύπου `woodstick` θα περιέχει 5.

Επιστρέφει

`DynamicArray<int>`

3.8.2.5 removeAfterCrafting()

```
DynamicArray< Item > Inventory::removeAfterCrafting (
    itemType itemType )
```

Επιστρέφει δυναμικό πίνακα που περιέχει τα πρωτογενή αντικείμενα που διαγράφηκαν για να κάνει craft ο χρήστης.

Εάν ο χρήστης για παραδειγμα δημιουργήσει ένα **Axe** από **Rock** και **woodstick** τότε θα προσθέσει τα 2 αντικείμενα αυτά στον δυναμικό πίνακα και μετά θα τα διαγράψει. Τέλος θα επιστεψει τον πίνακα.

Παράμετροι

| | |
|------------------------|--|
| itemCraftedType | Τύπος του παράγωγου αντικειμένου που ο χρήστης έκανε craft |
|------------------------|--|

Επιστρέφει

DynamicArray<Item> Δυναμικός Πίνακας που περιέχει τα πρωτογενή αντικείμενα που διαγράφηκαν.

3.8.2.6 removeAll()

```
void Inventory::removeAll ( )
```

Διαγράφει όλα τα αντικείμενα που βρίσκονται στον δυναμικό πίνακα

3.8.2.7 removeItemAt()

```
void Inventory::removeItemAt (
    int index )
```

Διαγράφει αντικείμενο στη συγκεκριμένη θέση που προσδιορίζεται από την παράμετρο index.

Παράμετροι

| | |
|--------------|--|
| index | |
|--------------|--|

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- Inventory/Inventory.h
- Inventory/Inventory.cpp

3.9 Τεκμηρίωση Κλάσης IoClass

Κλάση διαχείρισης εισοδου-εξόδου καθώς και διαχείρισης οθόνης.

```
#include <IoClass.h>
```

Δημόσιες Μέθοδοι

- `vector< Item > loadFromFile (string fileName, int &id)`
Διαβάζει το αρχείο και στη συνέχεια φορτώνει τα αντικείμενα σε *vector*.
- `void saveToFile (string fileName, Environment &env)`
Αντίστοιχα αποθηκεύει τα αντικείμενα που υπάρχουν στο *env*
Γενικά η γραμμογράφηση του αρχείου είναι ως εξής:
<itemId>;<itemName>;<itemTypeId>;<itemOnFloor>;<itemXpos>;<itemYpos>
- `string readString ()`
Διαβάζει *string* από το πληκτρολόγιο
- `void printToCoordsAnimated (int x, int y, string stringToPrint, std::initializer_list< string > a_args, int speed)`
Εμφανίζει σε συγκεκριμένες συντεταγμένες της οθόνης ένα αλφαριθμητικό καθώς και με προκαθορισμένη ταχύτητα
- `int showMenu (std::vector< string > selections)`
Δέχεται *vector* με *string* και εμφανίζει τα περιεχόμενα με μορφή μενού
- `void printEnvironment (Environment &env)`
Εμφανίζει το περιβάλλον(αντικείμενα, παίκτη κλπ)
- `void printPlayerStats (Player &player)`
Εμφάνιση στατιστικών του παίκτη(Επίπεδο πείνας)
- `~IoClass ()`
Destroy the Io Class object.
- `int getMaxX ()`
Getter για το μέγιστο X της οθόνης
- `int getMaxY ()`
Getter για το μέγιστο Y της οθόνης
- `int getInput ()`
Διαβάζει ένα χαρακτήρα από το πληκτρολόγιο και τον επιστρέφει

3.9.1 Λεπτομερής Περιγραφή

Κλάση διαχείρισης εισοδου-εξόδου καθώς και διαχείρισης οθόνης.

3.9.2 Τεκμηρίωση Constructor & Destructor

3.9.2.1 ~IoClass()

```
IoClass::~IoClass ( )
```

Destroy the Io Class object.

Destroy the Io Class:: Io Class object.

3.9.3 Τεκμηρίωση Συναρτήσεων Μελών

3.9.3.1 getInput()

```
int IoClass::getInput ( )
```

Διαβάζει ένανα χαρακτήρα απο το πληκτρολόγιο και τον επιστεφει

Επιστρέφει

int Χαρακτήρας που διαβάστηκε

3.9.3.2 getMaxX()

```
int IoClass::getMaxX ( )
```

Getter για το μέγιστο X της οθονης

Επιστρέφει

int μέγιστο X της οθονης

3.9.3.3 getMaxY()

```
int IoClass::getMaxY ( )
```

Getter για το μέγιστο Y της οθονης

Επιστρέφει

int μέγιστο Y της οθονης

3.9.3.4 loadFromFile()

```
vector< Item > IoClass::loadFromFile (
    string fileName,
    int & id )
```

Διαβάζει το αρχείο και στη συνέχεια φορτώνει τα αντικείμενα σε vector.

Παράμετροι

| | |
|-----------------|--|
| <i>fileName</i> | Το ονομα του αρχείου οπου θα προσπελάσει |
| <i>id</i> | Αναφορά σε ακέραιο αριθμό, χρησιμοποιείται για να υπολογίσει το τελευταίο id |

Επιστρέφει

`vector<Item> Container` όπου περιέχει τα αντικείμενα που διαβάστηκαν

3.9.3.5 printEnvironment()

```
void IoClass::printEnvironment (
    Environment & env )
```

Εμφανίζει το περιβάλλον(αντικείμενα, παίκτη κλπ)

Παράμετροι

| | |
|------------|--|
| <i>env</i> | |
|------------|--|

3.9.3.6 printPlayerStats()

```
void IoClass::printPlayerStats (
    Player & player )
```

Εμφάνιση στατιστικών του παίκτη(Επίπεδο πείνας)

Παράμετροι

| | |
|---------------|---|
| <i>player</i> | Αναφορά σε αντικείμενο Player |
|---------------|---|

3.9.3.7 printToCoordsAnimated()

```
void IoClass::printToCoordsAnimated (
    int x,
    int y,
    string stringToPrint,
    std::initializer_list< string > a_args,
    int speed )
```

Εμφανίζει σε συγκεκριμένες συντεταγμένες της οθόνης ένα αλφαριθμητικό καθώς και με προκαθορισμένη ταχύτητα

Παράμετροι

| | |
|----------|------------------|
| <i>x</i> | Η συντεταγμένη X |
| <i>y</i> | Η συντεταγμένη Y |

Παράμετροι

| | |
|----------------------|--|
| <i>stringToPrint</i> | Το αλφαριθμητικό που θα εμφανιστεί |
| <i>a_args</i> | Χρησιμοποιείται για να εμφανίζονται δυναμικά περιεχόμενα μεταβλητών(πχ όπως η printf έχει το s και το d) Οι μονες που υποστηρίζονται είναι οι s και d. |
| <i>speed</i> | Χρησιμοποιείται για την ταχύτητα εμφάνισης των χαρακτήρων. |

3.9.3.8 readString()

```
string IoClass::readString ( )
```

Διαβάζει string απο το πληκτρολόγιο

Επιστρέφει

string Το string που διαβάστηκε

3.9.3.9 saveToFile()

```
void IoClass::saveToFile (
    string fileName,
    Environment & env )
```

Αντίστοιχα αποθηκεύει τα αντικείμενα που υπάρχουν στο env

Γενικά η γραμμογράφηση του αρχείου είναι ως εξής:

<itemId>;<itemName>;<itemTypeId>;<itemOnFloor>;<itemXpos>;<itemYpos>

| itemId | itemName | itemTypeId | itemOnFloor | item x position | item y position |
|------------------------|---------------------------|----------------------------------|--------------------------------------|-----------------|-----------------|
| Το ID του αντικειμένου | Το όνομα του αντικειμένου | Το ID του τύπου του αντικειμένου | Flag για το εαν βρίσκεται στο πατωμα | Θέση X | Θέση Y |

Παράμετροι

| | |
|-----------------|---|
| <i>fileName</i> | Το ονομα του αρχείου οπου θα προσπελάσει |
| <i>env</i> | Αναφορά σε μεταβλητή τύπου Environment |

3.9.3.10 showMenu()

```
int IoClass::showMenu (
```

```
std::vector< string > selections )
```

Δέχεται **vector** με **string** και εμφανίζει τα περιεχόμενα με μορφή μενού

Παράμετροι

| | |
|-------------------------|--|
| <code>selections</code> | |
|-------------------------|--|

Επιστρέφει

`int` Το index της επιλογής που έγινε, 0 για την 1η επιλογή, 1 για την 2η επιλογή, 2 για την 3η κλπ..

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- IO/loClass.h
- IO/loClass.cpp

3.10 Τεκμηρίωση Κλάσης **Item**

Κλάση που χρησιμοποιείται για το κάθε αντικείμενο του παιχνιδιού.

```
#include <Item.h>
```

Διάγραμμα κληρονομικότητας για την **Item**:

Δημόσιες Μέθοδοι

- **Item** (string name, string id, **Vector2D**< int > position, bool isOnFloor=true)
- **Item** (const **Item** ©)
- string **getName** () const
- string **getId** () const
- bool **getIfOnFloor** () const
- itemType **getType** () const
- **Vector2D**< int > **getPosition** () const
- void **setType** (itemType type)
- void **setisOnFloor** (bool value)
- **Item operator+** (**Item** &item)

Χρησιμοποιείται για τον συνδιασμό/*crafting* δύο αντικειμένων σε ένα τρίτο αντικείμενο (παράγωγο των άλλων 2)

Σημείωση: Εάν αποτύχει τότε επιστρέφεται ένα αντικείμενο **Axe** με `id = null`

Ένας πίνακας που τα συνοψίζει είναι ο εξής:

3.10.1 Λεπτομερής Περιγραφή

Κλάση που χρησιμοποιείται για το κάθε αντικείμενο του παιχνιδιού.

3.10.2 Τεκμηρίωση Συναρτήσεων Μελών

3.10.2.1 operator+()

```
Item Item::operator+ (
    Item & item )
```

Χρησιμοποιείται για τον συνδιασμό/crafting δύο αντικειμένων σε ένα τρίτο αντικείμενο(παράγωγο των άλλων 2)

Σημείωση: Εάν αποτύχει τότε επιστρέφεται ένα αντικείμενο **Axe** με id = null

Ένας πίνακας που τα συνοψίζει είναι ο εξής:

| Item A | Item B | A + B |
|-----------|---------|--------------|
| WoodStick | Leafs | LightedTorch |
| Rock | Coconut | OpenCoconut |
| WoodStick | Rock | Axe |

Παράμετροι

| | |
|-------------|-------------------------------|
| <i>item</i> | Το αντικείμενο στο δεξί μέλος |
|-------------|-------------------------------|

Επιστρέφει

Item Το παράγωγο αντικείμενο που δημιουργήθηκε

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- Items/Item/Item.h
- Items/Item/Item.cpp

3.11 Τεκμηρίωση Κλάσης **Leafs**

Κλάση αντικειμένου **Leafs**

Χρησιμοποιείται για τα συμβολίσει Φύλλο

```
#include <Leafs.h>
```

Διάγραμμα κληρονομικότητας για την **Leafs**:

Διάγραμμα Συνεργασίας για την κλάση **Leafs**:

Δημόσιες Μέθοδοι

- **Leafs** (string name, string id, **Vector2D**< int >pos, bool isOnFloor=true)
- **Leafs** (const **Leafs** ©)

3.11.1 Λεπτομερής Περιγραφή

Κλάση αντικειμένου [Leafs](#)

Χρησιμοποιείται για τα συμβολίζει Φύλλο

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- Items/Leafs/Leafs.h
- Items/Leafs/Leafs.cpp

3.12 Τεκμηρίωση Κλάσης **LightedTorch**

Κλάση αντικειμένου [LightedTorch](#)

Χρησιμοποιείται για τα συμβολίζει αναμμένο πυρσό.

```
#include <LightedTorch.h>
```

Διάγραμμα κληρονομικότητας για την **LightedTorch**:

Διάγραμμα Συνεργασίας για την κλάση **LightedTorch**:

Δημόσιες Μέθοδοι

- **LightedTorch** (string name, string id, [Vector2D](#)< int >pos, bool isOnFloor=true)
- **LightedTorch** (const [LightedTorch](#) ©)

3.12.1 Λεπτομερής Περιγραφή

Κλάση αντικειμένου [LightedTorch](#)

Χρησιμοποιείται για τα συμβολίζει αναμμένο πυρσό.

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- Items/LightedTorch/LightedTorch.h
- Items/LightedTorch/LightedTorch.cpp

3.13 Τεκμηρίωση Κλάσης **Lighter**

Κλάση αντικειμένου [Lighter](#)

Χρησιμοποιείται για τα συμβολίζει αναπτήρα.

```
#include <Lighter.h>
```

Διάγραμμα κληρονομικότητας για την **Lighter**:

Διάγραμμα Συνεργασίας για την κλάση **Lighter**:

Δημόσιες Μέθοδοι

- **Lighter** (string name, string id, [Vector2D](#)< int >pos, bool isOnFloor=true)
- **Lighter** (const [Lighter](#) ©)

3.13.1 Λεπτομερής Περιγραφή

Κλάση αντικειμένου [Lighter](#)

Χρησιμοποιείται για τα συμβολίσει αναπτήρα.

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- Items/Lighter/Lighter.h
- Items/Lighter/Lighter.cpp

3.14 Τεκμηρίωση Κλάσης **OpenCoconut**

Κλάση αντικειμένου [OpenCoconut](#)

Χρησιμοποιείται για τα συμβολίσει ανοικτή καρύδα.

```
#include <OpenCoconut.h>
```

Διάγραμμα κληρονομικότητας για την [OpenCoconut](#):

Διάγραμμα Συνεργασίας για την κλάση [OpenCoconut](#):

Δημόσιες Μέθοδοι

- **OpenCoconut** (string name, string id, [Vector2D](#)< int >pos, bool isOnFloor=true)
- **OpenCoconut** (const [OpenCoconut](#) ©)

3.14.1 Λεπτομερής Περιγραφή

Κλάση αντικειμένου [OpenCoconut](#)

Χρησιμοποιείται για τα συμβολίσει ανοικτή καρύδα.

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- Items/OpenCoconut/OpenCoconut.h
- Items/OpenCoconut/OpenCoconut.cpp

3.15 Τεκμηρίωση Κλάσης **Player**

Κλάση για τον παίκτη του παιχνιδιού

```
#include <Player.h>
```

Δημόσιες Μέθοδοι

- **Player** (**Vector2D**< int > position, **Inventory** inv, string name="P")
Constructor της κλάσης Player.
- **Player** (const **Player** ©)
- void **setHunger** (int h)
- void **addToInventory** (**Item** &item)
 Προσθέτει αντικείμενο στον σάκο του χρήστη
- void **showInventory** (ostream &stream)
 Εμφανίζει σε *o/p stream* τα αντικείμενα του χρήστη
- int **getHunger** ()
 Επιστρέφει το επίπεδο τροφής του παίκτη
- string **getName** ()
Getter για το όνομα του χρήστη
- const **Vector2D**< int > & **getPosition** ()
Getter για την θέση του χρήστη
- **Inventory** & **getInventory** ()
Getter για τον σάκο με τα αντικείμενα του χρήστη
- **Item** & **getItemAt** (int index)
 Επιστρέφει αντικείμενο σε συγκεκριμένη θέση
- void **setName** (string newName)
- **DynamicArray**< **Item** > **getItems** ()
 Επιστρέφει δυναμικό πίνακα που περιέχει τα αντικείμενα του χρήστη
- void **moveToCoordinates** (int X, int Y)
 Μετακινεί τον χρήστη στις συντεταγμένες (X,Y)
- void **moveToCoordinates** (**Vector2D**< int > newPosition)
 Αντίστοιχη με την *moveToCoordinates(int X, int Y)* μόνο που χρησιμοποιεί *Vector2D<int>*
- void **removeFromInventory** (int index)
 Αφαιρεί αντικείμενο σε συγκεκριμένη θέση
- **Player operator=** (const **Player** ©)
- **DynamicArray**< **Item** > **getCrafted** ()
 Επιστρέφει δυναμικό πίνακα που περιέχει τα αντικείμενα που μπορεί ο χρήστης να κατασκευάσει
- void **removeAllFromPlayer** ()
 Καταστρέφει όλα τα αντικείμενα που βρίσκονται στον "σάκο" του χρήστη
- **DynamicArray**< **Item** > **removeAfterCrafting** (itemType itemType)
 Αφαιρεί τα αντικείμενα που χρησιμοποίησε ο χρήστης για να δημιουργήσει ένα νέο αντικείμενο και επιστρέφει δυναμικό πίνακα που τα περιέχει.

3.15.1 Λεπτομερής Περιγραφή

Κλάση για τον παίκτη του παιχνιδιού

3.15.2 Τεκμηρίωση **Constructor & Destructor**

3.15.2.1 **Player()**

```
Player::Player (
    Vector2D< int > position,
    Inventory inv,
    string name = "P" )
```

Constructor της κλάσης **Player**.

Παράμετροι

| | |
|-----------------|---|
| <i>position</i> | Vector2D που χρησιμοποιείται για την αρχική θέση του χρήστη |
| <i>inv</i> | Αντικείμενο Inventory που αρχικοποιεί τον σάκο του παίκτη |
| <i>name</i> | Το όνομα του παίκτη |

3.15.3 Τεκμηρίωση Συναρτήσεων Μελών

3.15.3.1 addToInventory()

```
void Player::addToInventory (
    Item & item )
```

Προσθέτει αντικείμενο στον σάκο του χρήστη

Παράμετροι

| | |
|-------------|--|
| <i>item</i> | |
|-------------|--|

3.15.3.2 getCrafted()

```
DynamicArray< Item > Player::getCrafted ( )
```

Επιστρέφει δυναμικό πίνακα που περιέχει τα αντικείμενα που μπορεί ο χρήστης να κατασκευάσει

Επιστρέφει

[DynamicArray](#)<[Item](#)>

3.15.3.3 getHunger()

```
int Player::getHunger ( ) [inline]
```

Επιστρέφει το επίπεδο τροφής του παίκτη

Επιστρέφει

int

3.15.3.4 `getInventory()`

```
Inventory& Player::getInventory ( ) [inline]
```

Getter για τον σάκο με τα αντικείμενα του χρήστη

Επιστρέφει

`Inventory&`

3.15.3.5 `getItemAt()`

```
Item & Player::getItemAt (
    int index )
```

Επιστρέφει αντικείμενο σε συγκεκριμένη θέση

Παράμετροι

| | |
|--------------|-------------------------|
| <i>index</i> | Η θέση του αντικειμένου |
|--------------|-------------------------|

Επιστρέφει

`Item&`

3.15.3.6 `getItems()`

```
DynamicArray< Item > Player::getItems ( )
```

Επιστρέφει δυναμικό πίνακα που περιέχει τα αντικείμενα του χρήστη

Επιστρέφει

`DynamicArray<Item>`

3.15.3.7 `getName()`

```
string Player::getName ( ) [inline]
```

Getter για το όνομα του χρήστη

Επιστρέφει

`string`

3.15.3.8 getPosition()

```
const Vector2D<int>& Player::getPosition ( ) [inline]
```

Getter για την θέση του χρήστη

Επιστρέφει

```
const Vector2D<int>&
```

3.15.3.9 moveToCoordinates() [1/2]

```
void Player::moveToCoordinates (
    int X,
    int Y )
```

Μετακινεί τον χρήστη στις συντεταγμένες (X,Y)

Παράμετροι

| | |
|---|--|
| X | |
| Y | |

3.15.3.10 moveToCoordinates() [2/2]

```
void Player::moveToCoordinates (
    Vector2D< int > newPosition )
```

Αντίστοιχη με την `moveToCoordinates(int X, int Y)` μόνο που χρησιμοποιεί `Vector2D<int>`

Παράμετροι

| | |
|--------------------|--|
| <i>newPosition</i> | |
|--------------------|--|

3.15.3.11 removeAfterCrafting()

```
DynamicArray< Item > Player::removeAfterCrafting (
    itemType itemCraftedType )
```

Αφαιρεί τα αντικείμενα που χρησιμοποίησε ο χρήστης για να δημιουργήσει ένα νέο αντικείμενο και επιστρέφει δυναμικό πίνακα που τα περιέχει.

Παράμετροι

| | |
|------------------------|--------------------------------------|
| <i>itemCraftedType</i> | Τύπος αντικειμένου που δημιουργήθηκε |
|------------------------|--------------------------------------|

Επιστρέφει

[DynamicArray<Item>](#) Δυναμικός πίνακας που περιέχει τα αντικείμενα που χρησιμοποίησε/κατέστρεψε ο χρήστης

3.15.3.12 removeAllFromPlayer()

```
void Player::removeAllFromPlayer ( )
```

Καταστρέφει όλα τα αντικείμενα που βρίσκονται στον "σακο" του χρήστη

3.15.3.13 removeFromInventory()

```
void Player::removeFromInventory (
    int index )
```

Αφαιρεί αντικείμενο σε συγκεκριμένη θέση

Παράμετροι

| | |
|--------------|--|
| <i>index</i> | |
|--------------|--|

3.15.3.14 showInventory()

```
void Player::showInventory (
    ostream & stream )
```

Εμφανίζει σε o/p stream τα αντικείμενα του χρήστη

Παράμετροι

| | |
|---------------|--|
| <i>stream</i> | |
|---------------|--|

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- Player/Player.h
- Player/Player.cpp

3.16 Τεκμηρίωση Κλάσης **Rock**

Κλάση αντικειμένου **Rock**

Χρησιμοποιείται για τα συμβολίζει πέτρα.

```
#include <Rock.h>
```

Διάγραμμα κληρονομικότητας για την **Rock**:

Διάγραμμα Συνεργασίας για την κλάση **Rock**:

Δημόσιες Μέθοδοι

- **Rock** (string name, string id, **Vector2D**< int >pos, bool isOnFloor=true)
- **Rock** (const **Rock** ©)

3.16.1 Λεπτομερής Περιγραφή

Κλάση αντικειμένου **Rock**

Χρησιμοποιείται για τα συμβολίζει πέτρα.

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- Items/Rock/Rock.h
- Items/Rock/Rock.cpp

3.17 Τεκμηρίωση Προτύπου Κλάσης **Vector2D< T >**

Κλάση για διάνυσμα δυοδιάστατου χώρου που είχα σχεδιάσει στο παρελθόν στα πλαίσια εξάσκησης για τα templates.

```
#include <Vector2D.h>
```

Δημόσιες Μέθοδοι

- **Vector2D** ()
Default Constructor αρχικοποιεί το X και Y σε 0,0.
- **Vector2D** (T x, T y)
Constructor που αρχικοποιεί τα X και Y με συγκεκριμένες τιμές
- **Vector2D** (const **Vector2D** &v)
Copy Constructor.
- **Vector2D** & operator= (const **Vector2D** &v)
Υπερφόρτωση τελεστή =.
- **Vector2D** operator+ (**Vector2D** &v)
Υπερφόρτωση τελεστή + ώστε να προσθέτει τα x και τα y των δύο αντικειμένων και επιστρέφει ένα καινούργιο που περιέχει αυτές τις τιμές
- **Vector2D** operator- (**Vector2D** &v)
Υπερφόρτωση τελεστή - αντιστοιχία του + ώστε να αφαιρεί

- **Vector2D & operator+= (Vector2D &v)**
Υπερφόρτωση τελεστή +=.
- **Vector2D & operator-= (Vector2D &v)**
Υπερφόρτωση τελεστή -=.
- **Vector2D operator+ (double s)**
Υπερφόρτωση τελεστή + με δεξί μέλος έναν αριθμό *double*.
- **Vector2D operator- (double s)**
Υπερφόρτωση τελεστή - με δεξί μέλος έναν αριθμό *double*.
- **Vector2D operator* (double s)**
Υπερφόρτωση τελεστή * με δεξί μέλος έναν αριθμό *double*.
- **Vector2D operator/ (double s)**
Υπερφόρτωση τελεστή / με δεξί μέλος έναν αριθμό *double*.
- **Vector2D & operator+= (double s)**
Υπερφόρτωση τελεστή += με δεξί μέλος έναν αριθμό *double*.
- **Vector2D & operator-= (double s)**
Υπερφόρτωση τελεστή -= με δεξί μέλος έναν αριθμό *double*.
- **Vector2D & operator*= (double s)**
Υπερφόρτωση τελεστή * με δεξί μέλος έναν αριθμό *double*.
- **Vector2D & operator/= (double s)**
Υπερφόρτωση τελεστή / με δεξί μέλος έναν αριθμό *double*.
- **void set (T x, T y)**
Setter και των δυο παραμετρών
- **void rotate (double deg)**
Χρησιμοποιείται για περιστροφή του διανύσματος κατά συγκεκριμένες μοίρες. Ο υπολογισμός των *X* και *Y* γίνεται με βάση γνωστό θεώρημα.
- **Vector2D & normalize ()**
Μεθοδος κανονικοποίησης διανύσματος. Για να κανονικοποιηθεί ένα διάνυσμα διαφείται το διάνυσμα με το μέτρο του. Έτσι προκύπτει ένα μοναδιαίο διάνυσμα. Δηλαδή δημιουργείται ένα διάνυσμα με μήκος ίσο με 1.
- **float length () const**
Επιστρέφει το μέτρο του διανύσματος

Δημόσια Χαρακτηριστικά

- **T x**
- **T y**

3.17.1 Λεπτομερής Περιγραφή

```
template<class T>
class Vector2D< T >
```

Κλάση για διάνυσμα δυσδιάστατου χώρου που είχα σχεδιάσει στο παρελθόν στα πλαίσια εξάσκησης για τα templates.

3.17.2 Τεκμηρίωση Constructor & Destructor

3.17.2.1 Vector2D() [1/3]

```
template<class T >
Vector2D< T >::Vector2D ( ) [inline]
```

Default Constructor αρχικοποιεί το X και Y σε 0,0.

3.17.2.2 Vector2D() [2/3]

```
template<class T >
Vector2D< T >::Vector2D (
    T x,
    T y ) [inline]
```

Constructor που αρχικοποιεί τα X και Y με συγκεκριμένες τιμές

Παράμετροι

| | |
|---|--|
| x | |
| y | |

3.17.2.3 Vector2D() [3/3]

```
template<class T >
Vector2D< T >::Vector2D (
    const Vector2D< T > & v ) [inline]
```

Copy Constructor.

Παράμετροι

| | |
|---|--|
| v | |
|---|--|

3.17.3 Τεκμηρίωση Συναρτήσεων Μελών**3.17.3.1 length()**

```
template<class T >
float Vector2D< T >::length ( ) const [inline]
```

Επιστρέφει το μέτρο του διανύσματος

Επιστρέφει

float Το μέτρο του διανύσματος

3.17.3.2 normalize()

```
template<class T >
Vector2D& Vector2D< T >::normalize ( ) [inline]
```

Μεθοδος κανονικοποίησης διανύσματος Για να κανονικοποιηθεί ένα διάνυσμα διαρείται το διάνυσμα με το μέτρο του. Έτσι προκύπτει ένα μοναδιαίο διάνυσμα. Δηλαδή δημιουργείται ένα διάνυσμα με μήκος ίσο με 1.

Επιστρέφει

Vector2D& Το διάνυσμα κανονικοποιημένο

3.17.3.3 operator*()

```
template<class T >
Vector2D Vector2D< T >::operator* (
    double s ) [inline]
```

Υπερφόρτωση τελεστη * με δεξί μέλος έναν αριθμό double.

Παράμετροι

| | |
|---|--|
| s | Ο αριθμος που θα πολλαπλασιαστεί με τα X και Y |
|---|--|

Επιστρέφει

Vector2D Αντικείμενο με τις νέες τιμές

3.17.3.4 operator*=()

```
template<class T >
Vector2D& Vector2D< T >::operator*= (
    double s ) [inline]
```

Υπερφόρτωση τελεστη * με δεξί μέλος έναν αριθμό double.

Παράμετροι

| | |
|----------|--|
| s | Ο αριθμός που θα πολλαπλασιαστεί με τα X και Y |
|----------|--|

Επιστρέφει

Vector2D Αντικείμενο με τις νέες τιμές

3.17.3.5 operator+() [1/2]

```
template<class T >
Vector2D Vector2D< T >::operator+ (
    double s ) [inline]
```

Υπερφόρτωση τελεστή + με δεξί μέλος έναν αριθμό double.

Παράμετροι

| | |
|----------|--|
| s | Ο αριθμός που θα προστεθεί στα X και Y |
|----------|--|

Επιστρέφει

Vector2D Αντικείμενο με τις νέες τιμές

3.17.3.6 operator+() [2/2]

```
template<class T >
Vector2D Vector2D< T >::operator+ (
    Vector2D< T > & v ) [inline]
```

Υπερφόρτωση τελεστή + ώστε να προσθέτει τα x και τα y των δύο αντικειμένων και επιστρέφει ένα καινούργιο που περιέχει αυτές τις τιμές

Παράμετροι

| | |
|----------|--|
| v | |
|----------|--|

Επιστρέφει

Vector2D

3.17.3.7 operator+=() [1/2]

```
template<class T >
Vector2D& Vector2D< T >::operator+= (
    double s ) [inline]
```

Υπερφόρτωση τελεστή += με δεξί μέλος εναν αριθμό double.

Παράμετροι

| | |
|----------|--|
| s | Ο αριθμος που θα προστεθεί στα X και Y |
|----------|--|

Επιστρέφει

Vector2D&

3.17.3.8 operator+=() [2/2]

```
template<class T >
Vector2D& Vector2D< T >::operator+= (
    Vector2D< T > & v ) [inline]
```

Υπερφόρτωση τελεστή +=.

Παράμετροι

| | |
|----------|--|
| v | |
|----------|--|

Επιστρέφει

Vector2D&

3.17.3.9 operator-() [1/2]

```
template<class T >
Vector2D Vector2D< T >::operator- (
    double s ) [inline]
```

Υπερφόρτωση τελεστή - με δεξί μέλος εναν αριθμό double.

Παράμετροι

| | |
|----------|---|
| s | Ο αριθμος που θα αφαιρεθεί απο τα X και Y |
|----------|---|

Επιστρέφει

Vector2D Αντικείμενο με τις νέες τιμές

3.17.3.10 operator-() [2/2]

```
template<class T >
Vector2D Vector2D< T >::operator- (
    Vector2D< T > & v ) [inline]
```

Υπερφόρτωση τελεστή - αντιστοιχία του + ώστε να αφαιρεί

Παράμετροι

| | |
|---|--|
| v | |
|---|--|

Επιστρέφει

Vector2D

3.17.3.11 operator-=() [1/2]

```
template<class T >
Vector2D& Vector2D< T >::operator-= (
    double s ) [inline]
```

Υπερφόρτωση τελεστή -= με δεξί μέλος έναν αριθμό double.

Παράμετροι

| | |
|---|---|
| s | Ο αριθμός που θα αφαιρεθεί από τα X και Y |
|---|---|

Επιστρέφει

Vector2D&

3.17.3.12 operator-=() [2/2]

```
template<class T >
Vector2D& Vector2D< T >::operator-= (
    Vector2D< T > & v ) [inline]
```

Υπερφόρτωση τελεστή -=.

Παράμετροι

| | |
|----------|--|
| <i>v</i> | |
|----------|--|

Επιστρέφει

[Vector2D](#)&

3.17.3.13 operator/()

```
template<class T >
Vector2D Vector2D< T >::operator/ (
    double s ) [inline]
```

Υπερφόρτωση τελεστη / με δεξί μέλος εναν αριθμό double.

Παράμετροι

| | |
|----------|---|
| <i>s</i> | Ο αριθμος που θα διαιρεθεί απο τα X και Y |
|----------|---|

Επιστρέφει

[Vector2D](#) Αντικείμενο με τις νέες τιμές

3.17.3.14 operator/=()

```
template<class T >
Vector2D& Vector2D< T >::operator/= (
    double s ) [inline]
```

Υπερφόρτωση τελεστη * με δεξί μέλος εναν αριθμό double.

Παράμετροι

| | |
|----------|---|
| <i>s</i> | Ο αριθμος που θα διαιρεθεί απο τα X και Y |
|----------|---|

Επιστρέφει

[Vector2D](#) Αντικείμενο με τις νέες τιμές

3.17.3.15 operator=()

```
template<class T >
Vector2D& Vector2D< T >::operator= (
    const Vector2D< T > & v ) [inline]
```

Υπερφόρτωση τελεστή =.

Παράμετροι

| | |
|----------|---|
| <i>v</i> | Το διάνυσμα απο το οποίο θα αντιγραφούν τα δεδομένα |
|----------|---|

Επιστρέφει

Vector2D&

3.17.3.16 rotate()

```
template<class T >
void Vector2D< T >::rotate (
    double deg ) [inline]
```

Χρησιμοποιείται για περιστροφή του διανύσματος κατα συγκεκριμένες μοίρες Ο υπολογισμός των X και Y γίνεται με βάση γνωστό θεώρημα.

Παράμετροι

| | |
|------------|--------|
| <i>deg</i> | Μοίρες |
|------------|--------|

3.17.3.17 set()

```
template<class T >
void Vector2D< T >::set (
    T x,
    T y ) [inline]
```

Setter και των δυο παραμετρων

Παράμετροι

| | |
|----------|--|
| <i>x</i> | |
| <i>y</i> | |

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από το ακόλουθο αρχείο:

- Vector2D/Vector2D.h

3.18 Τεκμηρίωση Κλάσης **WoodStick**

Κλάση αντικειμένου **WoodStick**

Χρησιμοποιείται για τα συμβολίζει ξύλο.

```
#include <WoodStick.h>
```

Διάγραμμα κληρονομικότητας για την **WoodStick**:

Διάγραμμα Συνεργασίας για την κλάση **WoodStick**:

Δημόσιες Μέθοδοι

- **WoodStick** (string name, string id, **Vector2D**< int >pos, bool isOnFloor=true)
- **WoodStick** (const **WoodStick** ©)

3.18.1 Λεπτομερής Περιγραφή

Κλάση αντικειμένου **WoodStick**

Χρησιμοποιείται για τα συμβολίζει ξύλο.

Η τεκμηρίωση για αυτή την κλάση δημιουργήθηκε από τα ακόλουθα αρχεία:

- Items/WoodStick/WoodStick.h
- Items/WoodStick/WoodStick.cpp

Index

- ~DynamicArray
 - DynamicArray< T >, 10
- ~Environment
 - Environment, 18
- ~IoClass
 - IoClass, 26
- addItem
 - Inventory, 23
- addItemToGround
 - Environment, 18
- addItemToInv
 - Environment, 19
- addToInventory
 - Player, 35
- Axe, 5
- begin
 - DynamicArray< T >, 10
- capacity
 - DynamicArray< T >, 11
- clear
 - DynamicArray< T >, 11
- Coconut, 5
- CSVRow, 6
 - operator[], 6
 - readNextRow, 6
 - size, 7
- DynamicArray
 - DynamicArray< T >, 9, 10
- DynamicArray< T >, 7
 - ~DynamicArray, 10
 - begin, 10
 - capacity, 11
 - clear, 11
 - DynamicArray, 9, 10
 - emplace_back, 11
 - end, 12
 - erase, 12
 - iterator, 8
 - move_storage, 12
 - operator=, 14
 - operator[], 14
 - pop_back, 14
 - push_back, 15
 - reallocate, 15
 - reference, 9
 - size, 15
 - size_type, 9
 - swap, 15
- emplace_back
 - DynamicArray< T >, 11
- end
 - DynamicArray< T >, 12
- Environment, 16
 - ~Environment, 18
 - addItemToGround, 18
 - addItemToInv, 19
 - Environment, 17, 18
 - generateGrid, 19
 - getItemsNearPlayer, 19
 - getPlayerCraft, 20
 - getPlayerItem, 20
 - handleLoadedData, 20
 - lastID, 21
 - movePlayer, 21
 - removeFromPlayerInv, 21
 - removeItemFromGround, 21
- erase
 - DynamicArray< T >, 12
- GameScene, 22
- generateGrid
 - Environment, 19
- getCrafted
 - Player, 35
- getHunger
 - Player, 35
- getInput
 - IoClass, 26
- getInventory
 - Player, 35
- getInventoryItems
 - Inventory, 24
- getItemAt
 - Inventory, 24
 - Player, 36
- getItems
 - Player, 36
- getItemsNearPlayer
 - Environment, 19
- getMaxX
 - IoClass, 27
- getMaxY
 - IoClass, 27
- getName
 - Player, 36

- getPlayerCraft
 - Environment, 20
- getPlayerItem
 - Environment, 20
- getPosition
 - Player, 36
- handleLoadedData
 - Environment, 20
- Hut, 22
- Inventory, 23
 - addItem, 23
 - getInventoryItems, 24
 - getItemAt, 24
 - itemTypeCount, 24
 - removeAfterCrafting, 24
 - removeAll, 25
 - removeItemAt, 25
- IoClass, 25
 - ~IoClass, 26
 - getInput, 26
 - getMaxX, 27
 - getMaxY, 27
 - loadFromFile, 27
 - printEnvironment, 28
 - printPlayerStats, 28
 - printToCoordsAnimated, 28
 - readString, 29
 - saveToFile, 29
 - showMenu, 29
- Item, 30
 - operator+, 31
- itemTypeCount
 - Inventory, 24
- iterator
 - DynamicArray< T >, 8
- lastID
 - Environment, 21
- Leafs, 31
- length
 - Vector2D< T >, 41
- LightedTorch, 32
- Lighter, 32
- loadFromFile
 - IoClass, 27
- move_storage
 - DynamicArray< T >, 12
- movePlayer
 - Environment, 21
- moveToCoordinates
 - Player, 37
- normalize
 - Vector2D< T >, 42
- OpenCoconut, 33
- operator*
 - Vector2D< T >, 42
- operator*=
 - Vector2D< T >, 42
- operator+
 - Item, 31
 - Vector2D< T >, 43
- operator+=
 - Vector2D< T >, 43, 44
- operator-
 - Vector2D< T >, 44, 45
- operator-=
 - Vector2D< T >, 45
- operator/
 - Vector2D< T >, 46
- operator/=
 - Vector2D< T >, 46
- operator=
 - DynamicArray< T >, 14
 - Vector2D< T >, 46
- operator[]
 - CSVRow, 6
 - DynamicArray< T >, 14
- Player, 33
 - addToInventory, 35
 - getCrafted, 35
 - getHunger, 35
 - getInventory, 35
 - getItemAt, 36
 - getItems, 36
 - getName, 36
 - getPosition, 36
 - moveToCoordinates, 37
 - Player, 34
 - removeAfterCrafting, 37
 - removeAllFromPlayer, 38
 - removeFromInventory, 38
 - showInventory, 38
- pop_back
 - DynamicArray< T >, 14
- printEnvironment
 - IoClass, 28
- printPlayerStats
 - IoClass, 28
- printToCoordsAnimated
 - IoClass, 28
- push_back
 - DynamicArray< T >, 15
- readNextRow
 - CSVRow, 6
- readString
 - IoClass, 29
- reallocate
 - DynamicArray< T >, 15
- reference
 - DynamicArray< T >, 9
- removeAfterCrafting
 - Inventory, 24

- Player, [37](#)
- removeAll
 - Inventory, [25](#)
- removeAllFromPlayer
 - Player, [38](#)
- removeFromInventory
 - Player, [38](#)
- removeFromPlayerInv
 - Environment, [21](#)
- removeItemAt
 - Inventory, [25](#)
- removeItemFromGround
 - Environment, [21](#)
- Rock, [39](#)
- rotate
 - Vector2D< T >, [47](#)
- saveToFile
 - IoClass, [29](#)
- set
 - Vector2D< T >, [47](#)
- showInventory
 - Player, [38](#)
- showMenu
 - IoClass, [29](#)
- size
 - CSVRow, [7](#)
 - DynamicArray< T >, [15](#)
- size_type
 - DynamicArray< T >, [9](#)
- swap
 - DynamicArray< T >, [15](#)
- Vector2D
 - Vector2D< T >, [40](#), [41](#)
- Vector2D< T >, [39](#)
 - length, [41](#)
 - normalize, [42](#)
 - operator*, [42](#)
 - operator*=[42](#)
 - operator+, [43](#)
 - operator+=, [43](#), [44](#)
 - operator-, [44](#), [45](#)
 - operator-=, [45](#)
 - operator/, [46](#)
 - operator/=, [46](#)
 - operator=, [46](#)
 - rotate, [47](#)
 - set, [47](#)
 - Vector2D, [40](#), [41](#)
- WoodStick, [48](#)