Growing Crystals

Последние изменения: 15.07.2014 17:35

Клиент

Переменные

gamestatus - **string,** хранит состояние игрового процесса для игрока, возможные значения:

- game - игровой процесс активен,
- menu - игрок находится в меню,
- startmenu - до запуска игры, во время ввода имени игрока
- load - загрузка игрового клиента

Протокол обмена данными

Клиент - Сервер

act - string

- n
- e
- s
- w
- start - если это значение то вместе с ним идёт имя пользователя
  player - string имя игрока
- c - если это значение тогда должно быть еще и
  const - **string** индекс объекта строительства
- up
- end

Сервер - Клиент

Описание

player - **object**, сущность игрока, хранит в себе все данные об игроке
name - **string** имя игрока

objectid - **int** id объекта игрока на карте
x - **int** абсолютная координата игрока на поле X увеличивается от W до S
y - **int** абсолютная координата игрока на поле Y увеличивается от N до E
z - **int** абсолютная координата игрока на поле Z пока не используется
sx - **int** screen X, абсолютная координата экрана игрока
sy - **int** screen Y, абсолютная координата экрана игрока
sz - **int** screen Z, абсолютная координата экрана игрока
cb - **int** Crystal Blue количество синих кристаллов игрока
cr - **int** Crystal Red количество красных кристаллов игрока
cy - **int** Crystal Yellow количество желтых кристаллов игрока
cbm - **int** Crystal Blue Max максимальная ёмкость синих кристаллов у игрока
crm - **int** Crystal Red Max максимальная ёмкость красных кристаллов у игрока
cym - **int** Crystal Yellow Max максимальная ёмкость жёлтых кристаллов у игрока
state - **string** содержит состояние игрока которое всегда сопадает с состоянием объекта игрока в игровом поле. Может принимать значения:

- anim - Обычное состояние
- construction - Состояние строительства

server - **object**,
ver - **string** версия сервера
timefromstart (tfs) - **float** секунды с начала игровой сессии
time - **string** строка времени сервера в формате HH:MM:SS

constructions - **array,** массив объектов содержит полное описание объектов строительства
…

produceevents - **array**, массив объектов - событий генерации кристаллов, событий окончания постройки и другие
…

objects - **array**, массив видимых объектов на поле
object, видимый объект
id - **int,** хранит id объекта
x,y,z - **int**, координаты объекта
movefrom - **object**, хранит координаты начала движения объекта
x - **int**
y - **int**
z - **int**
moveto - **object**, хранит координаты окончания движения объекта
x - **int**
y - **int**

z - **int**

anim - **object**, хранит анимацию объект состоит из двух объектов

move

onfinish - **string**, событие при завершении отрисовки анимаци значения:

- reset_state - время выполнения анимации сбросить на 0
- loop - взять остаток от деления на длительность анимации

frames - **array**, содержит номера спрайтов в анимации и их длительность в формате [[a1,a2], [b1,b2]] где a1, b1 - номер спрайтов a,b в файле с анимацией, a2, b2 - длительность показа спрайтов a,b соответственно. Если a2 или b2 == -1, спрайт a или b отрисовывается вечно.

sprite - **int**, номер строки спрайтов в файле с анимацией

loop

onfinish - **string**, событие при завершении отрисовки анимаци значения:

- reset_state - время выполнения анимации сбросить на 0
- loop - взять остаток от деления на длительность анимации

frames - **array**, содержит номера спрайтов в анимации и их длительность в формате [[x,y], [x,y]] где x - номер спрайта в файле с анимацией, y - длительность показа спрайта. Если y == -1, спрайт отрисовывается вечно.

sprite - **int**, номер строки спрайтов в файле с анимацией

Пример анимации для игрока:

```
anim: {
    move: {
        onfinish: 'reset_state',
        frames: [[0,200],[1,200],[3,100],[4,200],[5,200]],
        sprite: 1
    },
    loop: {
            onfinish: 'loop',
            frames: [[0,200],[1,200],[3,100],[4,200],[5,200]],
            sprite: 1
        }
    }
```

Пример анимации для бухающего чувака:

```
    anim: {
        move: {
            onfinish: 'loop',
            frames:
[[0,400],[1,300],[3,300],[4,300],[5,300],[6,300],[7,500],[6,200],[5,200],[4,200],[3,200],[2,200],[1,200],[0,3500]],
            sprite: 18
        },
        loop: {
```

```
                onfinish: 'loop',
                frames:
[[0,400],[1,300],[3,300],[4,300],[5,300],[6,300],[7,500],[6,200],[5,200],[4,200],[3,200],[2,200],[1,200],[0,3500]],
                sprite: 18
        }
    }
```

Пример анимации для статичного объекта:

```
    anim: {
        move: {
            onfinish: 'loop',
            frames: [[0,-1]],
            sprite: 4
        },
        loop: {
            onfinish: 'loop',
            frames: [[0,-1]],
            sprite: 4
        }
    }
```

Пример объекта столба:

```
{
    id: 111
    x: 1
    y: 1
    z: 1
    anim: {
        move: {
            onfinish: 'loop',
            frames: [[0,-1]],
            sprite: 14
        },
        loop: {
            onfinish: 'loop',
            frames: [[0,-1]],
            sprite: 14
        }
    }
}
```

Передаваемые при входе в игру

gamestatus
construtions
player
server
produceevents
objects

Передаваемые при перемещении игрока по полю и появлению новых объектов в области видимости

gamestatus
player {x,y,z,sx,sy,sz,cb,cr,cy,cbm,crm,cym, state}
server {timefromstart,time}
produceevents
objects

Передаваемые при всех остальных событиях

gamestatus
player {x,y,z,sx,sy,sz,cb,cr,cy,cbm,crm,cym}
server {timefromstart,time}
produceevents
objects {id,x,y,z}

Тонкости передачи Object

Когда создаётся новый объект, передаётся
objects {id,x,y,z,anim}

Когда объект меняет своё положение, передаётся
objects {id,x,y,z, movefrom, moveto}

Когда объект меняет своё состояние (state на сервере), передаётся
objects {id,x,y,z, anim}

Обычный режим передачи объектов
objects {id,x,y,z}