Dokidoki System Architecture

Author: kesumu

Created time: 2015-12-19

# Introduction

Dokidoki aimed at being a AVG development tool based on unity3d. It could improve the efficiency to develop AVG games and also could implement the one-time-code-multiple-platform-run based on the features of unity3d.

# Related projects

## Kirikiri

## NScript

# System architecture

## System

System class is responsible for arranging every thing inside dokidoki system. It contains a world which has lots of characters and one player. These characters take behaviors to interact with each other to progress the game. Behavior is a kind of action. Action is the minimum unit of things that happens in the AVG game, such as the position changing of a character, a sentence the character spoke. Effect is another kind of action such as the start of playing a sound, the start of the video and so on.

## Script

Scripts class is responsible to recognize the script written by AVG game developer, and then compile it into action sequences for System to conduct.

Example:

|  |
| --- |
| world  video src=video0;  bgm src=bgm0 mode=loop;  backgound src=background0 transition=instant;  weather type=snow level=0.2 transition=gradual speed=0.5;  天空渐渐飘下了雪花。>  在校门口隐约着有个人影。>  sound src=sound0;  我慢慢的走过去。>  dokiChan  role type=character name="character0";  move position=center transition=instant;  posture src=posture0;  face src=happyFace0;  voice src=voice001;  等你好久了。>>  face src=happyFace1;  voice src=voice002;  一直在等着你。>>  voice src=voice003;  还以为你不来了呢。>>  { //code in {} could not be skipped  world  我微笑着走了过去。>>  player  role type=player  face src=happyFace1;  怎么会呢，我们不是约好了么。>>  }  dokiChan  move position=(0.45,0,0) transition=instant; //Here, (0.45,0,0) could be 0.45  voice src=voice004;  是呢，一年前的约定。>>  player  嗯，一年过去了。>>    world  这家伙还是一点没变，仿佛时间已经抛弃了她。>>  两个人仅仅呆站这那儿，无言的看着对方。>>  weather type=sunny level=0.5 transition=gradual speed=0.5;  不经意间，雪停了>> |

|  |
| --- |
| dokidoki[D]:  P → program i ; D ; begin L end.  …to do  P → program i ; D ; begin L end.  D → var int i { , i }  A → i := E  E → T { + T }  T → F { \* F }  F → ( E ) | i  B → C { and M C}  C → H { or M H}  H → ( B ) | not B | E R E  R → > | < | <> | = | >= | <=  M → ε  S → A |if B then M S [ N else M S ] |while M B do M S | begin L end  L → S ; { M S ; }  N → ε |

# Requirements

Fasdf