Language Concept: DuelScript

Tagline: "It's Time to Code!"

DuelScript reimagines the structure and syntax of C++ through the lens of the Yu-Gi-Oh! trading card game. Every keyword and concept is replaced with iconic terms from the series, transforming programming from a task into a strategic duel.

Core Keyword Summoning

C++ Concept	DuelScript Keyword	Example
Program Start (main)	Yugi()	Ritual Yugi() { ... }
Class Definition	LordOfD	LordOfD Duelist { ... };
Struct Definition	ToonWorld	ToonWorld MonsterStats { ... };
Function Definition	Ritual	Ritual SummonDarkMagician() { ... }
Include	#SetField	#SetField <DuelArena>
Namespace	Kaiba	using Kaiba Joey;
Standard Namespace (std)	Joey	Joey::Summon << "Hello";
Output (cout)	Summon	Summon << "Blue-Eyes White Dragon, attack!";
Input (cin)	Draw	Draw >> lifePoints;
If Statement	JudgmentOfAnubis	JudgmentOfAnubis (attack > 1500) { ... }
Else Statement	SolemnJudgment	SolemnJudgment { ... }

While Loop	FairyBox	FairyBox (monsterAttack > 1000) { ... }
For Loop	SwordsOfRevealingLight	SwordsOfRevealingLight (DarkMagician i = 0; i < 3; i++) { ... }
Return	Tribute	Tribute 0;
Single-line Comment	MillenniumEye:	MillenniumEye: This activates my trap card.
Multi-line Comment	ShadowRealm{ ... }	ShadowRealm{ My grand strategy... }

Monster Data Types

Type	**Meaning**	**Example**
DarkMagician	Integer	DarkMagician attackPoints = 2500;
BlueEyesWhiteDragon	Double/Float	BlueEyesWhiteDragon powerLevel = 3000.5;
RedEyesBlackDragon	String	RedEyesBlackDragon monsterName = "Summoned Skull";
TimeWizard	Boolean	TimeWizard coinTossResult = true;

Syntax Examples

Struct Example (Toon World)

MillenniumEye: Create a structure to hold a monster's stats.

```
ToonWorld MonsterCard {
    RedEyesBlackDragon name;
    DarkMagician attack;
    DarkMagician defense;
};

Ritual Yugi() {
    MonsterCard darkMagician;
    darkMagician.name = "Dark Magician";
    darkMagician.attack = 2500;
    darkMagician.defense = 2100;

    Summon << "Card: " << darkMagician.name << ", ATK/" << darkMagician.attack;
}
```

Class Example (Lord of D.)

MillenniumEye: Define a class to manage a Duelist's state.

```
LordOfD Duelist {
  public: // Public Rituals and Monsters
    RedEyesBlackDragon name;
    DarkMagician lifePoints = 4000;

    Ritual Announce() {
        Summon << name << " has " << lifePoints << " Life Points remaining!\n";
    }
};

Ritual Yugi() {
    Duelist player1; // Summon a new Duelist object
    player1.name = "Yami Yugi";
    player1.Announce();
}
```

Conditional (Making a Judgment)

```
Ritual Yugi() {
    DarkMagician monsterAttack = 2100;

    JudgmentOfAnubis (monsterAttack > 3000) {
        Summon << "Your monster's attack is successful!";
    }
    SolemnJudgment {
        Summon << "Solemn Judgment! Your attack is negated!";
    }
}
```

Example Full Program (A Simple Duel)

```
#SetField <DuelArena>
using Kaiba Joey;

ToonWorld MonsterCard {
    RedEyesBlackDragon name;
    DarkMagician attack;
};

LordOfD Duelist {
  public:
    RedEyesBlackDragon name;
    DarkMagician lifePoints = 4000;

    Ritual TakeDamage(DarkMagician damage) {
        lifePoints -= damage;
        Summon << name << " takes " << damage << " points of damage!\n";
    }
};

Ritual Yugi() {
    ShadowRealm{
        This program simulates a simple turn-based attack
        using our custom class and struct.
    }

    Duelist player1;
    player1.name = "Yami Yugi";

    Duelist player2;
    player2.name = "Seto Kaiba";

    MonsterCard blueEyes;
    blueEyes.name = "Blue-Eyes White Dragon";
    blueEyes.attack = 3000;

    Summon << player1.name << " attacks " << player2.name << " with " << blueEyes.name <<
"!\n";
    player2.TakeDamage(blueEyes.attack);

    Summon << "Kaiba's remaining Life Points: " << player2.lifePoints << "\n";

    Tribute 0;
}
```