# 📘 标准文档模版：AI 题目解析 → DSL → JSON → 仿真引擎 → 动画 + 讲解

## 1. 概述

本文档定义了从「用户输入题目」到「生成动画和讲解」的完整处理流程。  
目标是实现一套 \*\*可扩展、可验证、跨学科\*\* 的教育动画生成系统。  
  
适用范围：  
- 学科：物理、化学、数学、生物  
- 学段：初一至高三  
- 应用场景：App/网页端互动学习

## 2. 流程总览

用户题目输入  
 │  
 ▼  
AI 解析 → 生成 DSL（符合系统设计规则）  
 │  
 ▼  
Parser 转换 DSL → JSON（符合 Schema）  
 │  
 ▼  
Validator 校验（补默认值 / 转类型 / 容错）  
 │  
 ▼  
仿真引擎执行（Matter.js / Three.js / JSmol / Plotly）  
 │  
 ├─ 动画渲染（小车 + 斜面 + 弹簧等物理场景）  
 └─ AI 并行生成讲解（公式推导 / 现象解释）

## 3. DSL 设计规范

DSL（Domain Specific Language）是领域专用描述语言，用于让 AI 将题目转化为结构化场景。  
  
### 示例：物理 DSL  
```  
Physics:  
 Object: cart1 mass=2 velocity=0 position=top\_of\_slope  
 Force: gravity g=9.8  
 Surface: slope angle=30 friction=0.2  
 Spring: k=100 length=1 attached\_to=cart1  
```

## 4. JSON Schema 规范

### 示例：物理场景 JSON  
```json  
{  
 "scene": "physics",  
 "objects": [  
 {  
 "id": "cart1",  
 "mass": 2,  
 "velocity": 0,  
 "position": "top\_of\_slope"  
 }  
 ],  
 "forces": {  
 "gravity": 9.8,  
 "friction": 0.2  
 },  
 "surface": {  
 "type": "slope",  
 "angle": 30  
 },  
 "spring": {  
 "k": 100,  
 "rest\_length": 1,  
 "attached\_to": "cart1"  
 },  
 "time": {  
 "start": 0,  
 "end": 10,  
 "step": 0.01  
 }  
}   
```

## 5. Validator（校验器）

Validator 用于保证 JSON 的正确性。  
- 校验必填字段是否存在  
- 校验类型是否正确  
- 自动补齐缺省值  
- 错误容错

## 6. 仿真引擎

仿真引擎负责执行 JSON 并生成动态结果。  
  
- 物理：Matter.js / Box2D / Ammo.js  
- 化学：JSmol / ChemDoodle  
- 数学：Plotly / Desmos API  
- 生物：细胞分裂 / DNA 模拟

## 7. 动画渲染

渲染层负责展示仿真结果：  
- Canvas / WebGL / Three.js 渲染  
- 支持箭头、力线、颜色标识  
- 用户可交互（暂停 / 拖拽 / 修改参数）

## 8. AI 讲解生成

并行调用 AI 生成文本讲解，包括：  
- 公式推导  
- 过程解释  
- 结论总结

## 9. 错误处理机制

AI 输出异常 DSL → Parser 抛错  
JSON 校验失败 → Validator 补齐默认值或报错  
引擎无法绘制对象 → 适配器回退到简化模型

## 10. 长期演进方案

短期：AI → DSL → JSON → 仿真引擎  
中期：引入少量微调，让 AI 更稳定地产生 DSL  
长期：DSL → JSON → 引擎 标准库；支持跨学科组合题目