11711BCC008
Matheus José da Costa

Na criação de um jogo, sabemos que é primordial a implementação de diversos recursos aos quais serão utilizados para fazer movimentação de elementos, sons e afins.

Suponhamos um jogo em que tenha um único personagem principal, e o objetivo principal é ultrapassar a linha de chegada. Só que, para isso, há obstáculos no mapa, aos quais ele pode colidir. Dessa forma, precisamos fazer uma programação específica para quando o personagem colidir com a caixa, e então, dar o famoso “game over”. Da mesma forma, precisamos implementar a física do personagem, pois senão, a cada pulo que ele dá, para passar por cima da caixa, pode ir muito alto ou muito baixo. Ou seja, precisamos calibrar tudo isso.

Dessa forma, fazendo a programação de um estilo de jogo como esse, temos muitas coisas que podemos reutilizar. Principalmente relacionado a física.

Então, se quisermos, posteriormente, criar um novo jogo, com uma nova temática, porém com traços físicos semelhantes, poderíamos, sim, utilizar essa mesma engine para abstrair todo esse complexo anteriormente descrito. Assim, poupando tempo de desenvolvimento.

Para isso, temos diversos padrões de projetos aos quais podemos utilizar para facilitar esse reuso. Dentre eles, podemos citar a tentativa de desencapsular o código, para uma menor dependência.

Além disso, utilizar o princípio de Inversão de Dependência será muito valioso, pois nos permitirá desaclopar classes concretas; e com isso também temos que não poderemos derivar classes de classes concretas, mas sim de abstrações.

E preocupar-se com interfaces, ao invés de implementações, também gerará benefícios. Pois nos preocuparemos mais com as abstrações ao invés da lógica de fato, indo em direção ao uso do polimorfismo.

Mantermos nossas classes suscetíveis a extensão, porém a não modificação, garantirá que, num futuro, caso tenha necessidade de modificar um comportamento, não precisaremos mudar o código atual, mas sim criarmos uma nova classe e extendermos ela.

Com essas explicações, vemos que esses princípios podem ser utilizados e muito bem na criação de uma Engine. Porque precisamos de algo melhor utilizável e passível a mudanças – em caso de uma mudança de regra.