



图形用户界面, 文本, 应用程序

AI 生成的内容可能不正确。

那么，从头开始的话，Object系统有那么多功能：GC，反射，[序列化](https://zhida.zhihu.com/search?content_id=1964795&content_type=Article&match_order=1&q=%E5%BA%8F%E5%88%97%E5%8C%96&zhida_source=entity)，[编辑器支持](https://zhida.zhihu.com/search?content_id=1964795&content_type=Article&match_order=1&q=%E7%BC%96%E8%BE%91%E5%99%A8%E6%94%AF%E6%8C%81&zhida_source=entity)……应该从哪一个开始？哪一个是必需的？GC不是，因为大不了我可以直接new/delete或者智能指针引用技术，毕竟别的很多引擎也都是这么干的。序列化也不是，大不了每个子类里手写数据排布，麻烦是麻烦，但是功能上也是可以实现的。编辑器支持，默认类对象，统计等都是之后额外附加的功能了。那你说反射为何是必需的？大多数游戏引擎用的C++没有反射，不也都用得好好的？确实也如此，不利用反射的那些功能，不动态根据类型创建对象，不遍历属性成员，不根据名字调用函数，大不了手写绕一下，没有过不去的坎。但是既然上文已经论述了一个统一Object模型的好处，那么如果在Object身上不加上反射，无疑就像是砍掉了Object的一双翅膀，让它只能在地上行走，而不能在更宽阔空间内发挥威力。还有另一个方面的考虑是，反射作为底层的系统，如果实现完善了，也可以大大裨益其他系统的实现，比如有了反射，实现序列化起来就很方便了；有没有反射，也关系到GC实现时的方案选择，完全是两种套路。简单举个例，反射里对每个object有个class对象保存信息，所以理论上class身上就可以保存所有该类型的object指针引用，这个信息GC就可以利用起来实现一些功能；而没有这个class对象的话，GC的实现就得走别的方案路子了。所以说是先实现反射，有了一个更加扎实的对象系统基础后，再在此之上实现GC才更加的明智。

图形用户界面, 文本, 应用程序

AI 生成的内容可能不正确。

图形用户界面, 文本, 应用程序, 电子邮件

AI 生成的内容可能不正确。

图形用户界面, 文本

AI 生成的内容可能不正确。



图示

AI 生成的内容可能不正确。

图示

AI 生成的内容可能不正确。