Awesome-qr.js 类耦合问题分析

注：仅列出人工判定或程序判定中至少一个为真的条目, CINT 阈值 FEW=3

类名>函数名	人工/程序判定	DC 强度	FE 强度	人工/程序判定耦合对象	说明	开发者意见
QRPolynomial>mod	是/是	0.4	0	QRMath/QRMath 及入参 e	应该放进 QRMath 里？	存在耦合。QRPolynomial 应该是与 QR 二维码标准中的 ErrorCorrectionLevel 编码相关的类，QRMath 在初始化时生成 LOG_TABLE 和 EXP_TABLE 之后便将其作为常量列表使用。因 QRPolynomial 存在递归调用，观察得到 QRMath 只会被 QRPolynomial 调用，可以把 QRMath 的部分放入 QRPolynomial 直接使用。
QRPolynomial>multiply	是/是	0.5	0	QRMath/QRMath 及入参 e	同上。	同上，应将孤立无其他类调用的 QRMath 放入 QRPolynomial。
Drawing>draw	是/是	0.07	0	QRCodeModel 等多个类/ QRUtil, QRCodeModel	函数是 Blob，代码体积大，内聚性低，耦合并非其主要问题。	该函数在编写完毕后维护起来较为困难，将每个处理步骤拆分成将 options 作为参数传入的函数较好。
QRCodeModel>make	是/是	0.33	0.17(QRRSBlock)	QRUtil / QRRSBlock, QRUtil	将部分逻辑移动到工具类。	检查时发现函数体中存在无用代码，改进时可以直接调用 makeImpl 函数。
AwesomeQRCode>makeCode	是/是	0.4	1(Drawing, QRCodeModel)	Drawing, QRCodeModel	检测通过调用推断_oDrawing 是 Drawing 类的实例。	AwesomeQRCode 旨在将 Drawing 与 QRCodeModel 的操作包装起来，并对外隐藏掉部分不必要的接口，其对外暴露的接口会尽可能地避免改动。
QRUtil>getLostPoint	是/否	NA	NA	QRCodeModel	QRUtil 未被检测工具检测为类。	该函数对于选定容错级别下的 mask patterns 进行测试并选出最优（所需变动最小）的一个测试结果并以其确定二维码绘制矩阵。

Awesome-qr.js 类内聚问题分析

注：列出所有类

类名	人工/程序判定	结构方式强度	文本方式强度	行数	说明	开发者意见
QRCodeModel	是/是	0.576	0.51	533	体积庞大。	将二维码编码、读取元数据信息等方法整合至同一类下，存储当前二维码的元数据信息，一定程度上也充当了一个逻辑流的 wrapper。
Drawing	是/是	0.68	0.74	327	体积庞大。	draw 函数所占行数较多，应进行拆分。
QRPolynomial	是/是	0	0.29	44	该类有工具类的特征。	与 QRMath 合并后应该会提高一些内聚度。
QRBitBuffer	是/是	0.26	0.21	28	定义了较多的成员（属性、方法），但在内部的访问次数较少，于成员个数相同，但问题不严重。	用于存储容错编码前处理之前的原始二维码数据流，因此只在创建二维码时较早地调用。
QR8bitByte	是/是	1	0	41	因为类中定义了多个未经函数成员访问的属性，内聚性不足，但函数成员的名称却与业务逻辑有关，故仅结构方式报告了问题。	将输入数据编码为每元素 8-byte 的数组，构造函数即完成了初始化流。
AwesomeQRCode	是/是	0	0.46	55	仅文本方式报告了问题，显示其涉及的概念多，这类情况通常出现在较大的初始化函数中。	作为内部类与方法调用的 wrapper，将编码渲染工作流整合至一个函数下。
QRRSBlock	否/否	0	0	197	类本身较精简，但用字面变量定义了一个很大的数组，所以行数看起来很大。	与容错编码 Reed–Solomon 相关。
QRUtil	是/否	NA	NA	213	未检测到类。	用于存储 mask patterns 以及提供选出最佳 pattern 的依据（penalty），近似常量的数组由函数暴露给外部根据 typeNumber 访问。
QRMath	否/否	NA	NA	18	未检测到类。	该类应被移除并合并至 QRPolynomial 下。