com.radar

**Class FileParser**

public class FileParser

负责解析头文件信息、读取数据并写入到图像中。

头文件信息包括Dimension、Variable和Attribute三个部分，Variable自身也包含有Dimension和Attribute。

雷达数据会写入彩色图像，卫星数据写入黑白图像。

*方法摘要：*

RadarFile readWithImg(String fileIn, String imagePath)

解析文件的入口方法。

Data4Json readGridData(String filePath)

读取路径中的文件数据，返回Data4Json对象

*方法详细信息:*

**readWithImg**

public RadarFile readWithImg( String fileIn,String name )

打开传入的文件，调用readHeadInfo方法解析其头文件，读取数据并生成数据的图像。返回一个RadarFile对象，包含存储有头文件信息的RadarHeadfile对象、图片类型、数据和图片路径。

参数：fileIn - 文件的完整路径（包括文件名）

imagePath - 生成的图片的存储路径

异常：IOException - 如果文件无法读取。

**readGridData**

public Data4Json readGridData(String filePath)

打开传入的文件，读取其数据。传入的文件须为Grid格式。该方法通过RasterGridBuilder对象完成具体工作。

com.radar

**Class ImageCreator**

Public class ImageCreator

*常量成员：*

public final static int GRAY

黑白Grid数据

public final static int RGB\_GRID

彩色Grid数据

public final static int RGB\_RADIAL

彩色Radial数据

*方法摘要：*

String createImage(int type,

int[] data,int width,int height, String fileName)

将笛卡尔直角坐标系数据写入图像，返回图像文件路径。

String createImage(int type,

float[] data,float[] azimuth,int gate,String fileName)

将极坐标数据写入图像，返回图像文件路径。

void setImagePath(String imagePath)

设置图片存储的目录

**createImage**

public String writeRGBFile(int[] data,int width,int height,String fileName)

使用BufferedImage对象创建图像，type允许为GRAY和RGB\_GRID，逐像素绘图；使用ImageIO以png格式输出。

参数：

type - 图片绘制类型

data - 存储整型数据的数组

width - 图片宽度

height - 图片高度

fileName - 图片名称

异常：IOException - 如果图片文件无法创建或写入。

**createImage**

public String writePixFile(int[] data,int width,int height,String fileName)

使用BufferedImage对象创建图像，type允许为RGB\_RADIAL，逐像素绘图；使用ImageIO以png格式输出。

参数：

type - 图片绘制类型

data - 存储浮点数据的数组

azimuth - 方位角数据

gate - 通道数，表示每条半径线上的数据量

fileName - 图片名称

异常：IOException - 如果图片文件无法创建或写入。

**setImagePath**

public void setImagePath(String imagePath)

在解析文件前可以设置图片的存储目录（不包含文件名），在绘图时使用该路径。默认为空，图片存储在项目根目录。

com.radar.radargrid

**Class RasterGridBuilder**

Public class RasterGridBuilder

负责读取Grid格式数据，将数据的起始x、y，格网单元大小、行数、列数和数据的2维数组存储在RasterGrid2\_Byte对象中返回。

*方法摘要：*

RasterGrid2\_Byte build(Variable var, List<CoordinateAxis)

根据传入的数据集读取数据，返回RasterGrid2\_Byte对象

*方法详细信息:*

**build**

public RasterGrid2\_Byte build(Variable var, List<CoordinateAxis)

接收包含数据的Variable对象和对应的坐标轴，返回一个RasterGrid2\_Byte对象。支持Grid格式的数据。源数据读取时会强制转换成byte类型。

com.radar.radial

**class RadarSweep**

public class RadarSweep implements Data4Json

为Radial数据传输设计的数据模型

*构造方法：*

RadarSweep(float[] m\_data, float[] azimuth, int gNum, float gSize, float gRange)

m\_data：数据所在数组

azimuth：方位角数据的数组

gNum：gate number，表示一条径线上的数据数量

gSize：gate的长度（米）

gRange：中心点到第一个gate的距离（一般为0？）；

com.radar.radial

**Class RadarSweepBuilder**

public class RadarSweepBuilder

用于创建RadarSweep对象

*方法摘要：*

RadarSweep build(RadialDatasetSweep.RadialVariable varRef)

根据传入的RadialVariable，读取、返回一个sweep范围的数据

com.radar

**Class RadarFile**

存储头文件信息和数据图像地址的POJO

*方法摘要：*

void setHeadfile(RadarHeadfile headfile)

头文件对象的Setter方法

void setImgType(int imgType)

void setImgData(Data4Json imgData)

void setImgUrl(String imgUrl) 图片地址的Setter方法

RadarHeadfile getHeadfile()

int getImgType()

Data4Json getImgData()

String getImgUrl()

com.radar

**Class RadarFile**

存储头文件信息的POJO

*方法摘要：*

void setDimention(Map dimention)

void setVariable(Map variable)

void setAttribute(Map attribute)

void setBounds(double[][] bounds)

bounds是一个二维数组double[2][2]，表示数据经纬度范围

Map<String, String> getDimention()

Map<String, String> getAttribute()

Map<String, Variable4Json> getVariable()

double[][] getBounds()

com.radar

**Class Variable4Json**

为适应Structure数据创建，存储变量类型、属性和成员。在转换JSON时没有内容的字段不会转换。

*方法摘要：*

void setType(String type)

void setAttribute(String attribute)

void setMembers(Map members)

String getType()

String getAttribute()

Map getMembers()

