# Macsur Adapter

Generated by Doxygen 1.8.3.1

Tue Jun 4 2013 12:58:10

# Contents

# Chapter 1

# Namespace Index

## 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

# Chapter 2

# Hierarchical Index

## 2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

# Chapter 3

# Class Index

## 3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

# Chapter 4

# File Index

## 4.1 File List

Here is a list of all files with brief descriptions:

# Chapter 5

# Namespace Documentation

## 5.1 Ui Namespace Reference

# Chapter 6

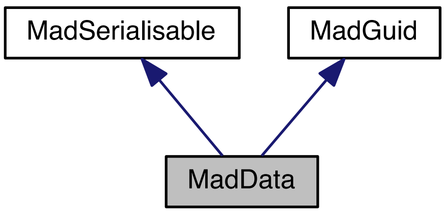
# Class Documentation

## 6.1 MadData Class Reference

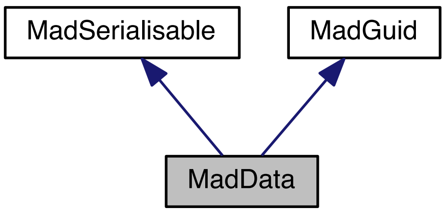
The MadData class.

`#include <maddata.h>`

Inheritance diagram for MadData:



Collaboration diagram for MadData:



### Public Member Functions

- MadData ()
- MadData (const MadData &theData)
- MadData & operator= (const MadData &theData)
- QString name () const

  *name (accessor) this is the dataset's name*
- QString description () const

  *description (accessor) this is the dataset's description*
- QString imageFile () const
- void setName (QString theName)
- void setDescription (QString theDescription)
- void setImageFile (QString theImageFileName)
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.1.1 Detailed Description

The MadData class.

Definition at line 38 of file maddata.h.

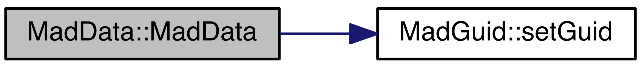
### 6.1.2 Constructor & Destructor Documentation

#### 6.1.2.1 MadData::MadData ( )

Definition at line 32 of file maddata.cpp.

```
                   : MadSerialisable(), MadGuid()
{
    setGuid();
    mName="No Name Set";
    mDescription="Not Set";
}
```

Here is the call graph for this function:



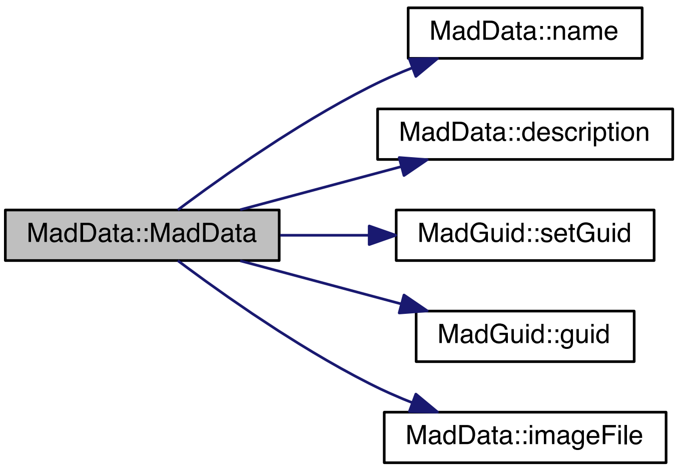
#### 6.1.2.2 MadData::MadData ( const MadData & *theData* )

copy constructor

Definition at line 39 of file maddata.cpp.

```
{
    mName=theData.name();
    mDescription=theData.description();
    setGuid(theData.guid());
    mImageFile=theData.imageFile();
}
```

Here is the call graph for this function:



### 6.1.3 Member Function Documentation

#### 6.1.3.1 QString MadData::description ( ) const

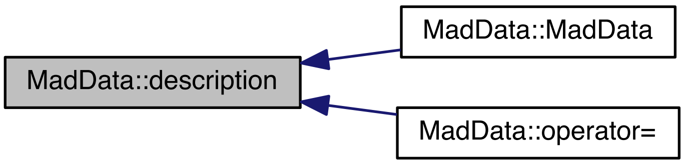
description (accessor) this is the dataset's description

**Returns**

Definition at line 65 of file maddata.cpp.

```
{
    return mDescription;
}
```

Here is the caller graph for this function:



### 6.1.3.2 bool MadData::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

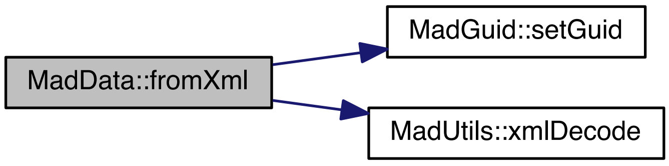
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 94 of file maddata.cpp.

```
{
    QDomDocument myDocument("mydocument");
    myDocument.setContent(theXml);
    QDomElement myTopElement = myDocument.firstChildElement("model");
    if (myTopElement.isNull())
     {
         //TODO - just make this a warning
         qDebug("the top element couldn't be found!");
         setGuid(myTopElement.attribute("guid"));
         mName=MadUtils::xmlDecode(myTopElement.firstChildElement("name").text());
         mDescription=MadUtils::xmlDecode(myTopElement.firstChildElement("description").
      text());
         mImageFile=QString(myTopElement.firstChildElement("imageFile").text());
         return true;
     }
     else
     return false;
}
```

Here is the call graph for this function:



#### 6.1.3.3 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) `[virtual], [inherited]`

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

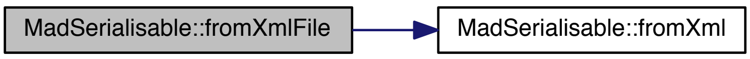
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



#### 6.1.3.4 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

```
{
    return mGuid;
}
```

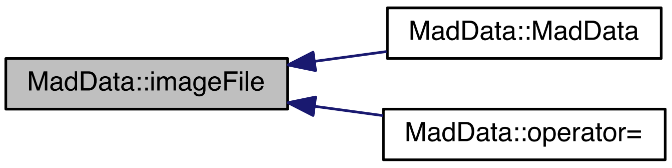
#### 6.1.3.5 QString MadData::imageFile ( ) const

The image file associated with the dataset

Definition at line 70 of file maddata.cpp.

```
{
    return mImageFile;
}
```

Here is the caller graph for this function:



#### 6.1.3.6 QString MadData::name ( ) const

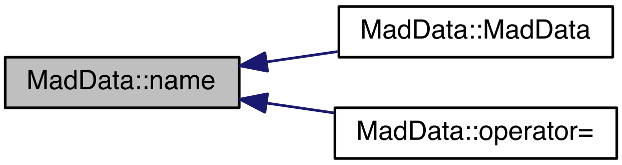
name (accessor) this is the dataset's name

**Returns**

Definition at line 60 of file maddata.cpp.

```
{
    return mName;
}
```

Here is the caller graph for this function:



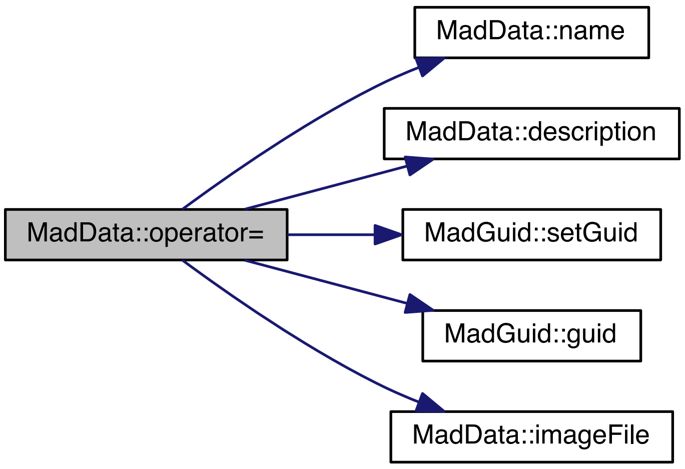
#### 6.1.3.7 MadData & MadData::operator= ( const MadData & *theData* )

Assignement operator

Definition at line 47 of file maddata.cpp.

```
{
    if (this == &theData) return *this; // gracefully handles self assignment

    mName=theData.name();
    mDescription=theData.description();
    setGuid(theData.guid());
    mImageFile=theData.imageFile();
    return *this;
}
```

Here is the call graph for this function:



#### 6.1.3.8 void MadData::setDescription ( QString *theDescription* )

Set the model description

**See Also**

description()

Definition at line 82 of file maddata.cpp.

```
{
    mDescription=theDescription;
}
```

#### 6.1.3.9 void MadGuid::setGuid ( QString *theGuid* = "" ) [inherited]

MadGuid::setGuid.

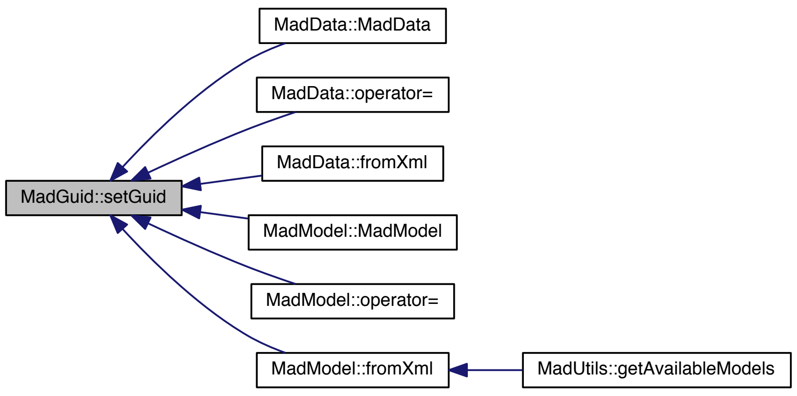
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:



#### 6.1.3.10 void MadData::setImageFile ( QString *theImageFileName* )

Set the image file

**See Also**

imageFile()

Definition at line 87 of file maddata.cpp.

```
{
    mImageFile=theImageFileName;
}
```

#### 6.1.3.11 void MadData::setName ( QString *theName* )

Set the modelName

**See Also**

name()

Definition at line 77 of file maddata.cpp.

```
{
    mName=theName;
}
```

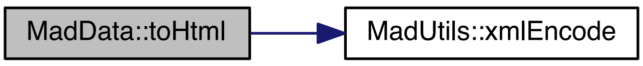
#### 6.1.3.12 QString MadData::toHtml ( )

Return a html text representation of this layer

Definition at line 133 of file maddata.cpp.

```
{
  QString myString;
  myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

Here is the call graph for this function:



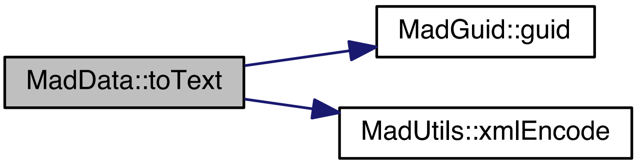
#### 6.1.3.13 QString MadData::toText ( )

Return a plain text representation of this layer

Definition at line 124 of file maddata.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.1.3.14 QString MadData::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

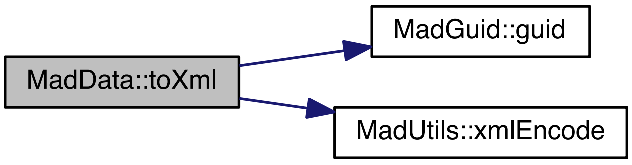
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 113 of file maddata.cpp.

```
{
  QString myString;
  myString+=QString("<dataset guid=\"" + guid() + "\">\n");
  myString+=QString("  <name>" + MadUtils::xmlEncode(mName) + "</name>\n");
  myString+=QString("  <description>" + MadUtils::xmlEncode(mDescription) + "
  </description>\n");
  myString+=QString("  <imageFile>" + MadUtils::xmlEncode(mImageFile) + "</imageFile>\n"
  );
  myString+=QString("</dataset>\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.1.3.15 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) `[virtual], [inherited]`

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

|  |  |
|---|---|
| *theFileName* |  |

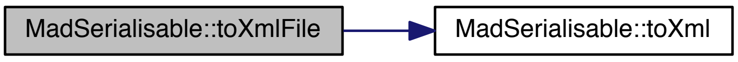
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



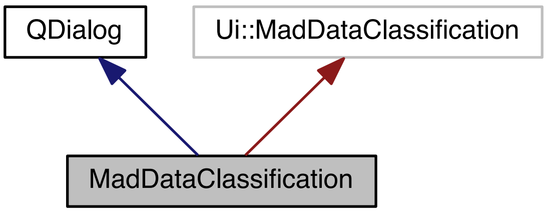
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddata.h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddata.cpp

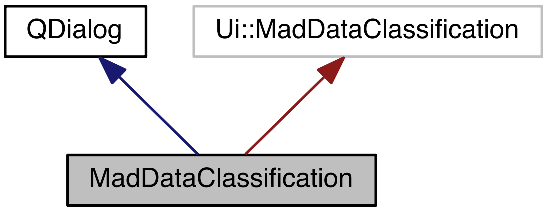
## 6.2 MadDataClassification Class Reference

`#include <maddataclassification.h>`

Inheritance diagram for MadDataClassification:



Collaboration diagram for MadDataClassification:



### Public Member Functions

- MadDataClassification (QWidget ∗parent=0)

### Protected Member Functions

- void changeEvent (QEvent ∗e)

## 6.2.1 Detailed Description

Definition at line 38 of file maddataclassification.h.

## 6.2.2 Constructor & Destructor Documentation

### 6.2.2.1 MadDataClassification::MadDataClassification ( QWidget ∗ *parent* = 0 ) [explicit]

Definition at line 33 of file maddataclassification.cpp.

```
                                                        : QDialog(parent)
{
    setupUi(this);
```

```
gbxCultivation->setChecked(false);
cbExamples->setEnabled(true);
lblExample->setVisible(true);
lblExample->setText("Select Example");
lblMedalCultivation->setVisible(false);
lblRankingCultivation->setVisible(false);
lblExample->setVisible(true);
cbExamples->setDisabled(false);

gbxPhenology->setChecked(false);
cbExamplesPhenology->setEnabled(true);
lblExamplePhenology->setVisible(true);
lblExamplePhenology->setText("Select Example");
lblMedalPhenology->setVisible(false);
lblRankingPhenology->setVisible(false);
lblExamplePhenology->setVisible(true);
cbExamplesPhenology->setDisabled(false);

// These must stay here at the end

// cultivation connections
connect ( sbVariety, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateVarietyRatingLbl() ));
connect ( dsbVariety, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateVarietyRatingLbl() ));
connect ( sbSowing, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSowingRatingLbl() ));
connect ( dsbSowing, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSowingRatingLbl() ));
connect ( sbHarvest, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateHarvestRatingLbl() ));
connect ( dsbHarvest, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateHarvestRatingLbl() ));
connect ( sbFertilisation, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateFertilisationRatingLbl() ));
connect ( dsbFertilisation, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateFertilisationRatingLbl() ));
connect ( sbIrrigation, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateIrrigationRatingLbl() ));
connect ( dsbIrrigation, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateIrrigationRatingLbl() ));
connect ( sbSeedDensity, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSeedDensityRatingLbl() ));
connect ( dsbSeedDensity, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSeedDensityRatingLbl() ));
connect ( sbYield, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateYieldRatingLbl() ));
connect ( dsbYield, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateYieldRatingLbl() ));
connect ( sbTillage, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateTillageRatingLbl() ));
connect ( dsbTillage, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateTillageRatingLbl() ));

// phenology connections
connect ( sbEmergencePhenology, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updatePhenologyEmergenceRatingLbl() ));
connect ( dsbEmergencePhenology, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updatePhenologyEmergenceRatingLbl() ));
connect ( sbStemElongationPhenology, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updatePhenologyStemElongationRatingLbl() ));
connect ( dsbStemElongationPhenology, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updatePhenologyStemElongationRatingLbl() ));
connect ( sbEarEmergencePhenology, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updatePhenologyEarEmergenceRatingLbl() ));
connect ( dsbEarEmergencePhenology, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updatePhenologyEarEmergenceRatingLbl() ));
connect ( sbFloweringPhenology, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updatePhenologyFloweringRatingLbl() ));
connect ( dsbFloweringPhenology, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updatePhenologyFloweringRatingLbl() ));
connect ( sbYellowRipenessPhenology, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updatePhenologyYellowRipenessRatingLbl() ));
connect ( dsbYellowRipenessPhenology, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updatePhenologyYellowRipenessRatingLbl() ));

// prev crop connections
connect ( sbCropPrevCrop, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updatePrevCropCropRatingLbl() ));
connect ( dsbCropPrevCrop, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updatePrevCropCropRatingLbl() ));
connect ( sbSowingDatePrevCrop, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updatePrevCropSowingDateRatingLbl() ));
connect ( dsbSowingDatePrevCrop, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updatePrevCropSowingDateRatingLbl() ));
connect ( sbHarvestDatePrevCrop, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updatePrevCropHarvestDateRatingLbl() ));
```

```
connect ( dsbHarvestDatePrevCrop, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updatePrevCropHarvestDateRatingLbl() ));
connect ( sbYieldPrevCrop, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updatePrevCropYieldRatingLbl() ));
connect ( dsbYieldPrevCrop, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updatePrevCropYieldRatingLbl() ));
connect ( sbResidueMgmtPrevCrop, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updatePrevCropResidueMgmtRatingLbl() ));
connect ( dsbResidueMgmtPrevCrop, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updatePrevCropResidueMgmtRatingLbl() ));
connect ( sbFertilisationPrevCrop, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updatePrevCropFertilisationRatingLbl() ));
connect ( dsbFertilisationPrevCrop, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updatePrevCropFertilisationRatingLbl() ));
connect ( sbIrrigationPrevCrop, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updatePrevCropIrrigationRatingLbl() ));
connect ( dsbIrrigationPrevCrop, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updatePrevCropIrrigationRatingLbl() ));

// initial values connections
connect ( sbSoilMoistureInitialValues, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateInitialValuesSoilMoistureRatingLbl() ));
connect ( dsbSoilMoistureInitialValues, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateInitialValuesSoilMoistureRatingLbl() ));
connect ( sbNMinInitialValues, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateInitialValuesNMinRatingLbl() ));
connect ( dsbNMinInitialValues, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateInitialValuesNMinRatingLbl() ));

// soil connections
connect ( sbCOrgSoil, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSoilCOrgRatingLbl() ));
connect ( dsbCOrgSoil, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSoilCOrgRatingLbl() ));
connect ( sbNOrgSoil, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSoilNOrgRatingLbl() ));
connect ( dsbNOrgSoil, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSoilNOrgRatingLbl() ));
connect ( sbTextureSoil, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSoilTextureRatingLbl() ));
connect ( dsbTextureSoil, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSoilTextureRatingLbl() ));
connect ( sbBulkDensitySoil, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSoilBulkDensityRatingLbl() ));
connect ( dsbBulkDensitySoil, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSoilBulkDensityRatingLbl() ));
connect ( sbFieldCapacitySoil, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSoilFieldCapacityRatingLbl() ));
connect ( dsbFieldCapacitySoil, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSoilFieldCapacityRatingLbl() ));
connect ( sbWiltingPointSoil, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSoilWiltingPointRatingLbl() ));
connect ( dsbWiltingPointSoil, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSoilWiltingPointRatingLbl() ));
connect ( sbPfCurveSoil, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSoilPfCurveRatingLbl() ));
connect ( dsbPfCurveSoil, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSoilPfCurveRatingLbl() ));
connect ( sbHydrCondCurveSoil, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSoilHydrCondCurveRatingLbl() ));
connect ( dsbHydrCondCurveSoil, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSoilHydrCondCurveRatingLbl() ));
connect ( sbPhSoil, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSoilPhRatingLbl() ));
connect ( dsbPhSoil, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSoilPhRatingLbl() ));

// site data connections
connect ( sbLatitudeSite, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSiteLatitudeRatingLbl() ));
connect ( dsbLatitudeSite, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSiteLatitudeRatingLbl() ));
connect ( sbLongitudeSite, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSiteLongitudeRatingLbl() ));
connect ( dsbLongitudeSite, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSiteLongitudeRatingLbl() ));
connect ( sbAltitudeSite, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSiteAltitudeRatingLbl() ));
connect ( dsbAltitudeSite, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSiteAltitudeRatingLbl() ));

// weather connections
connect ( sbPrecipitationWeather, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateWeatherPrecipitationRatingLbl() ));
connect ( dsbPrecipitationWeather, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateWeatherPrecipitationRatingLbl() ));
connect ( sbTAveWeather, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateWeatherTAveRatingLbl() ));
connect ( dsbTAveWeather, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateWeatherTAveRatingLbl() ));
connect ( sbTMinWeather, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateWeatherTMinRatingLbl() ));
connect ( dsbTMinWeather, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateWeatherTMinRatingLbl() ));
connect ( sbTMaxWeather, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateWeatherTMaxRatingLbl() ));
connect ( dsbTMaxWeather, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateWeatherTMaxRatingLbl() ));
connect ( sbRelHumidityWeather, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateWeatherRelHumidityRatingLbl() ));
connect ( dsbRelHumidityWeather, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateWeatherRelHumidityRatingLbl() ));
connect ( sbWindSpeedWeather, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateWeatherWindSpeedRatingLbl() ));
connect ( dsbWindSpeedWeather, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateWeatherWindSpeedRatingLbl() ));
connect ( sbGlobalRadiationWeather, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateWeatherGlobalRadiationRatingLbl() ));
connect ( dsbGlobalRadiationWeather, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateWeatherGlobalRadiationRatingLbl() ));
connect ( sbSunshineHoursWeather, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateWeatherSunshineHoursRatingLbl() ));
connect ( dsbSunshineHoursWeather, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateWeatherSunshineHoursRatingLbl() ));
connect ( sbLeafWetnessWeather, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateWeatherLeafWetnessRatingLbl() ));
connect ( dsbLeafWetnessWeather, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateWeatherLeafWetnessRatingLbl() ));
connect ( sbSoilTempWeather, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateWeatherSoilTempRatingLbl() ));
connect ( dsbSoilTempWeather, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateWeatherSoilTempRatingLbl() ));

// state vars connections
//crop
connect ( dsbSVCropAGrBiomassLayers, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropAGrBiomassRatingLbl() ));
connect ( sbSVCropAGrBiomassObservations, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateSVCropAGrBiomassRatingLbl() ));
connect ( dsbSVCropAGrBiomassWeightPts, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropAGrBiomassRatingLbl() ));
connect ( dsbSVCropAGrBiomassReplicates, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropAGrBiomassRatingLbl() ));

connect ( dsbSVCropWeightOrgansLayers, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropWeightOrgansRatingLbl() ));
connect ( sbSVCropWeightOrgansObservations, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateSVCropWeightOrgansRatingLbl() ));
connect ( dsbSVCropWeightOrgansWeightPts, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropWeightOrgansRatingLbl() ));
connect ( dsbSVCropWeightOrgansReplicates, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropWeightOrgansRatingLbl() ));

connect ( dsbSVCropRootBiomassLayers, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropRootBiomassRatingLbl() ));
connect ( sbSVCropRootBiomassObservations, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateSVCropRootBiomassRatingLbl() ));
connect ( dsbSVCropRootBiomassWeightPts, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropRootBiomassRatingLbl() ));
connect ( dsbSVCropRootBiomassReplicates, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropRootBiomassRatingLbl() ));

connect ( dsbSVCropNInAGrBiomassLayers, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropNInAGrBiomassRatingLbl() ));
connect ( sbSVCropNInAGrBiomassObservations, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateSVCropNInAGrBiomassRatingLbl() ));
connect ( dsbSVCropNInAGrBiomassWeightPts, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropNInAGrBiomassRatingLbl() ));
connect ( dsbSVCropNInAGrBiomassReplicates, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropNInAGrBiomassRatingLbl() ));

connect ( dsbSVCropNInOrgansLayers, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropNInOrgansRatingLbl() ));
connect ( sbSVCropNInOrgansObservations, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateSVCropNInOrgansRatingLbl() ));
connect ( dsbSVCropNInOrgansWeightPts, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropNInOrgansRatingLbl() ));
connect ( dsbSVCropNInOrgansReplicates, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropNInOrgansRatingLbl() ));

connect ( dsbSVCropLAILayers, SIGNAL ( valueChanged(double) ),
            this, SLOT ( updateSVCropLAIRatingLbl() ));
connect ( sbSVCropLAIObservations, SIGNAL ( valueChanged(int) ),
            this, SLOT ( updateSVCropLAIRatingLbl() ));
```

```
connect ( dsbSVCropLAIWeightPts, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVCropLAIRatingLbl() ));
connect ( dsbSVCropLAIReplicates, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVCropLAIRatingLbl() ));

// soil
connect ( dsbSVSoilSoilWaterSensorCalLayers, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilSoilWaterSensorCalRatingLbl() ));
connect ( sbSVSoilSoilWaterSensorCalObservations, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSVSoilSoilWaterSensorCalRatingLbl() ));
connect ( dsbSVSoilSoilWaterSensorCalWeightPts, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilSoilWaterSensorCalRatingLbl() ));
connect ( dsbSVSoilSoilWaterSensorCalReplicates, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilSoilWaterSensorCalRatingLbl() ));

connect ( dsbSVSoilPressureHeadsLayers, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilPressureHeadsRatingLbl() ));
connect ( sbSVSoilPressureHeadsObservations, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSVSoilPressureHeadsRatingLbl() ));
connect ( dsbSVSoilPressureHeadsWeightPts, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilPressureHeadsRatingLbl() ));
connect ( dsbSVSoilPressureHeadsReplicates, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilPressureHeadsRatingLbl() ));

connect ( dsbSVSoilNMinLayers, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilNMinRatingLbl() ));
connect ( sbSVSoilNMinObservations, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSVSoilNMinRatingLbl() ));
connect ( dsbSVSoilNMinWeightPts, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilNMinRatingLbl() ));
connect ( dsbSVSoilNMinReplicates, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilNMinRatingLbl() ));

connect ( dsbSVSoilSoilWaterSensorCalLayers, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilSoilWaterSensorCalRatingLbl() ));
connect ( sbSVSoilSoilWaterSensorCalObservations, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSVSoilSoilWaterSensorCalRatingLbl() ));
connect ( dsbSVSoilSoilWaterSensorCalWeightPts, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilSoilWaterSensorCalRatingLbl() ));
connect ( dsbSVSoilSoilWaterSensorCalReplicates, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilSoilWaterSensorCalRatingLbl() ));

connect ( dsbSVSoilWaterFluxBottomRootLayers, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilWaterFluxBottomRootRatingLbl() ));
connect ( sbSVSoilWaterFluxBottomRootObservations, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSVSoilWaterFluxBottomRootRatingLbl() ));
connect ( dsbSVSoilWaterFluxBottomRootWeightPts, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilWaterFluxBottomRootRatingLbl() ));
connect ( dsbSVSoilWaterFluxBottomRootReplicates, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilWaterFluxBottomRootRatingLbl() ));

connect ( dsbSVSoilNFluxBottomRootLayers, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilNFluxBottomRootRatingLbl() ));
connect ( sbSVSoilNFluxBottomRootObservations, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSVSoilNFluxBottomRootRatingLbl() ));
connect ( dsbSVSoilNFluxBottomRootWeightPts, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilNFluxBottomRootRatingLbl() ));
connect ( dsbSVSoilNFluxBottomRootReplicates, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSoilNFluxBottomRootRatingLbl() ));

// surface fluxes
connect ( dsbSVSurfaceFluxesEtLayers, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSurfaceFluxesEtRatingLbl() ));
connect ( sbSVSurfaceFluxesEtObservations, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSVSurfaceFluxesEtRatingLbl() ));
connect ( dsbSVSurfaceFluxesEtWeightPts, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSurfaceFluxesEtRatingLbl() ));
connect ( dsbSVSurfaceFluxesEtReplicates, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSurfaceFluxesEtRatingLbl() ));

connect ( dsbSVSurfaceFluxesNh3LossLayers, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSurfaceFluxesNh3LossRatingLbl() ));
connect ( sbSVSurfaceFluxesNh3LossObservations, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSVSurfaceFluxesNh3LossRatingLbl() ));
connect ( dsbSVSurfaceFluxesNh3LossWeightPts, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSurfaceFluxesNh3LossRatingLbl() ));
connect ( dsbSVSurfaceFluxesNh3LossReplicates, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSurfaceFluxesNh3LossRatingLbl() ));

connect ( dsbSVSurfaceFluxesN2OLossLayers, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSurfaceFluxesN2OLossRatingLbl() ));
connect ( sbSVSurfaceFluxesN2OLossObservations, SIGNAL ( valueChanged(int) ),
          this, SLOT ( updateSVSurfaceFluxesN2OLossRatingLbl() ));
connect ( dsbSVSurfaceFluxesN2OLossWeightPts, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSurfaceFluxesN2OLossRatingLbl() ));
connect ( dsbSVSurfaceFluxesN2OLossReplicates, SIGNAL ( valueChanged(double) ),
          this, SLOT ( updateSVSurfaceFluxesN2OLossRatingLbl() ));
```

```

    connect ( dsbSVSurfaceFluxesN2LossLayers, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVSurfaceFluxesN2LossRatingLbl() ));
    connect ( sbSVSurfaceFluxesN2LossObservations, SIGNAL ( valueChanged(int) ),
              this, SLOT ( updateSVSurfaceFluxesN2LossRatingLbl() ));
    connect ( dsbSVSurfaceFluxesN2LossWeightPts, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVSurfaceFluxesN2LossRatingLbl() ));
    connect ( dsbSVSurfaceFluxesN2LossReplicates, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVSurfaceFluxesN2LossRatingLbl() ));

    connect ( dsbSVSurfaceFluxesCh4LossLayers, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVSurfaceFluxesCh4LossRatingLbl() ));
    connect ( sbSVSurfaceFluxesCh4LossObservations, SIGNAL ( valueChanged(int) ),
              this, SLOT ( updateSVSurfaceFluxesCh4LossRatingLbl() ));
    connect ( dsbSVSurfaceFluxesCh4LossWeightPts, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVSurfaceFluxesCh4LossRatingLbl() ));
    connect ( dsbSVSurfaceFluxesCh4LossReplicates, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVSurfaceFluxesCh4LossRatingLbl() ));

    // observations
    connect ( dsbSVObservationsLodgingLayers, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVObservationsLodgingRatingLbl() ));
    connect ( sbSVObservationsLodgingObservations, SIGNAL ( valueChanged(int) ),
              this, SLOT ( updateSVObservationsLodgingRatingLbl() ));
    connect ( dsbSVObservationsLodgingWeightPts, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVObservationsLodgingRatingLbl() ));
    connect ( dsbSVObservationsLodgingReplicates, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVObservationsLodgingRatingLbl() ));

    connect ( dsbSVObservationsPestsOrDiseasesLayers, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVObservationsPestsOrDiseasesRatingLbl() ));
    connect ( sbSVObservationsPestsOrDiseasesObservations, SIGNAL ( valueChanged(int) ),
              this, SLOT ( updateSVObservationsPestsOrDiseasesRatingLbl() ));
    connect ( dsbSVObservationsPestsOrDiseasesWeightPts, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVObservationsPestsOrDiseasesRatingLbl() ));
    connect ( dsbSVObservationsPestsOrDiseasesReplicates, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVObservationsPestsOrDiseasesRatingLbl() ));

    connect ( dsbSVObservationsDamagesLayers, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVObservationsDamagesRatingLbl() ));
    connect ( sbSVObservationsDamagesObservations, SIGNAL ( valueChanged(int) ),
              this, SLOT ( updateSVObservationsDamagesRatingLbl() ));
    connect ( dsbSVObservationsLodgingWeightPts, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVObservationsDamagesRatingLbl() ));
    connect ( dsbSVObservationsLodgingReplicates, SIGNAL ( valueChanged(double) ),
              this, SLOT ( updateSVObservationsDamagesRatingLbl() ));
  }
```

### 6.2.3 Member Function Documentation

#### 6.2.3.1 void MadDataClassification::changeEvent ( QEvent * e ) [protected]

Definition at line 431 of file maddataclassification.cpp.

```
  {
    QDialog::changeEvent(e);
    switch (e->type()) {
    case QEvent::LanguageChange:
      retranslateUi(this);
      break;
    default:
      break;
    }
  }
```

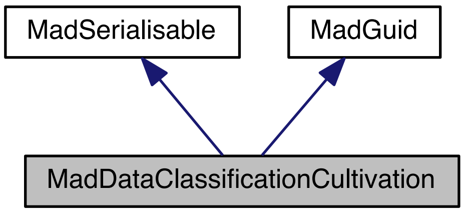
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/maddataclassification.h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/maddataclassification.cpp

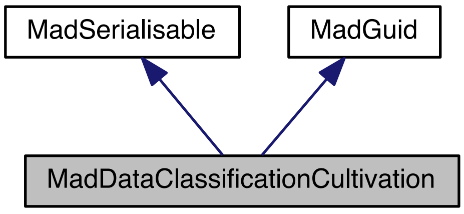
## 6.3 MadDataClassificationCultivation Class Reference

`#include <maddataclassificationcultivation.h>`

Inheritance diagram for MadDataClassificationCultivation:



Collaboration diagram for MadDataClassificationCultivation:



## Public Member Functions

- MadDataClassificationCultivation ()
- MadDataClassificationCultivation (const MadDataClassificationCultivation &theData)
- MadDataClassificationCultivation & operator= (const MadDataClassificationCultivation &theData)
- MadSubCategory variety () const
- MadSubCategory sowing () const
- MadSubCategory harvest () const
- MadSubCategory fertilisation () const
- MadSubCategory irrigation () const
- MadSubCategory seedDensity () const
- MadSubCategory yield () const
- MadSubCategory tillage () const
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- void setVariety (MadSubCategory theData)
- void setSowing (MadSubCategory theData)
- void setHarvest (MadSubCategory theData)
- void setFertilisation (MadSubCategory theData)
- void setIrrigation (MadSubCategory theData)

- void setSeedDensity (MadSubCategory theData)
- void setYield (MadSubCategory theData)
- void setTillage (MadSubCategory theData)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.3.1 Detailed Description

Definition at line 35 of file maddataclassificationcultivation.h.

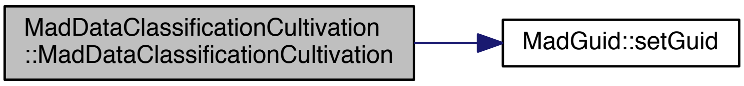
### 6.3.2 Constructor & Destructor Documentation

#### 6.3.2.1 MadDataClassificationCultivation::MadDataClassificationCultivation ( )

Definition at line 33 of file maddataclassificationcultivation.cpp.

```
                                                                   :
     MadSerialisable(), MadGuid()
{
    setGuid();
}
```

Here is the call graph for this function:

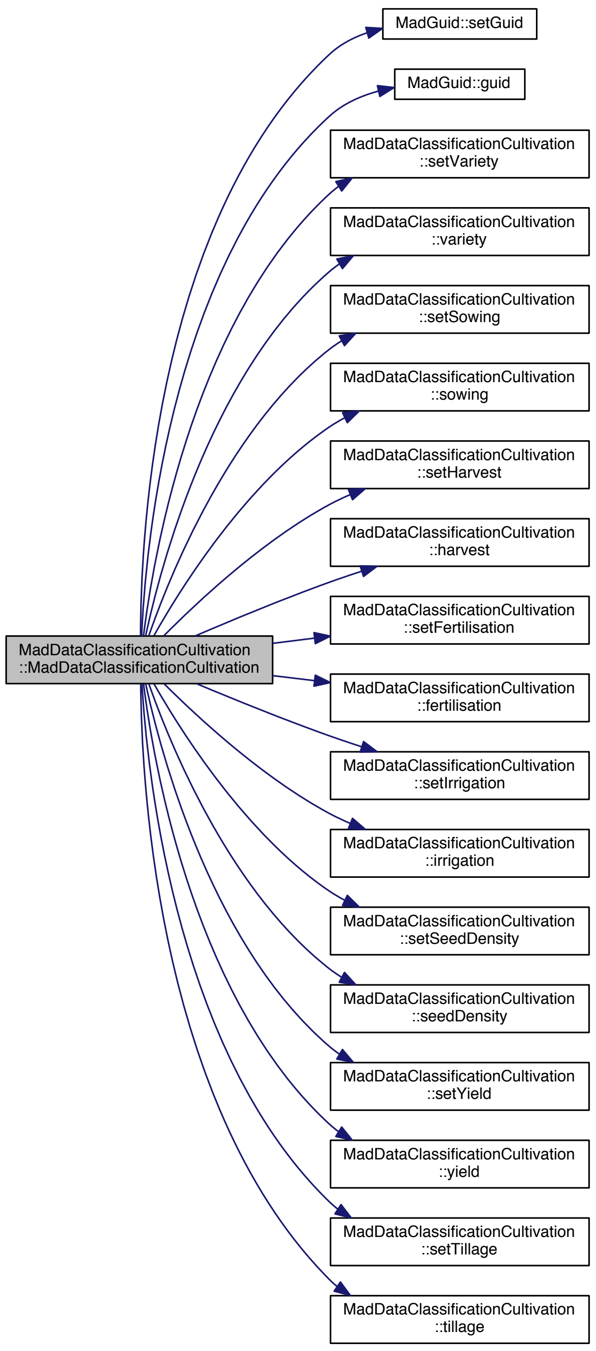

MadDataClassificationCultivation ::MadDataClassificationCultivation
MadGuid::setGuid

#### 6.3.2.2 MadDataClassificationCultivation::MadDataClassificationCultivation ( const MadDataClassificationCultivation & *theData* )

Definition at line 38 of file maddataclassificationcultivation.cpp.

```
{
    setGuid(theData.guid());
    setVariety(theData.variety());
    setSowing(theData.sowing());
    setHarvest(theData.harvest());
    setFertilisation(theData.fertilisation());
    setIrrigation(theData.irrigation());
    setSeedDensity(theData.seedDensity());
    setYield(theData.yield());
    setTillage(theData.tillage());
}
```

Here is the call graph for this function:


MadDataClassificationCultivation ::MadDataClassificationCultivation
MadGuid::setGuid
MadGuid::guid
MadDataClassificationCultivation ::setVariety
MadDataClassificationCultivation ::variety
MadDataClassificationCultivation ::setSowing
MadDataClassificationCultivation ::sowing
MadDataClassificationCultivation ::setHarvest
MadDataClassificationCultivation ::harvest
MadDataClassificationCultivation ::setFertilisation
MadDataClassificationCultivation ::fertilisation
MadDataClassificationCultivation ::setIrrigation
MadDataClassificationCultivation ::irrigation
MadDataClassificationCultivation ::setSeedDensity
MadDataClassificationCultivation ::seedDensity
MadDataClassificationCultivation ::setYield
MadDataClassificationCultivation ::yield
MadDataClassificationCultivation ::setTillage
MadDataClassificationCultivation ::tillage

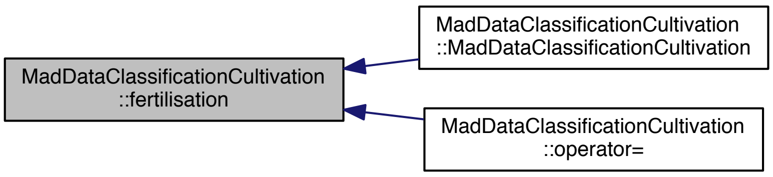
### 6.3.3 Member Function Documentation

#### 6.3.3.1 MadSubCategory MadDataClassificationCultivation::fertilisation ( ) const

Definition at line 80 of file maddataclassificationcultivation.cpp.

```
81 {
82     return mFertilisation;
}
```

Here is the caller graph for this function:


MadDataClassificationCultivation ::fertilisation
MadDataClassificationCultivation ::MadDataClassificationCultivation
MadDataClassificationCultivation ::operator=

#### 6.3.3.2 bool MadDataClassificationCultivation::fromXml ( const QString *theXml* ) [virtual]

true for success, false for failure.

**See Also**

MadSerialisable

**Note**

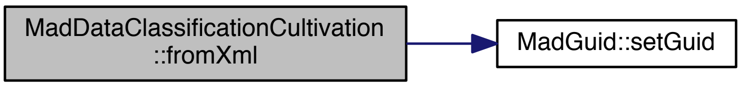
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 136 of file maddataclassificationcultivation.cpp.

```
{
    QDomDocument myDocument("mydocument");
    myDocument.setContent(theXml);
    QDomElement myTopElement = myDocument.firstChildElement("cultivation");
    if (myTopElement.isNull())
    {
        //TODO - just make this a warning
        qDebug("the top element couldn’t be found!");
        setGuid(myTopElement.attribute("guid"));
        //mName=MadUtils::xmlDecode(myTopElement.firstChildElement("name").text());
        //mDescription=MadUtils::xmlDecode(myTopElement.firstChildElement("description").text());
        //mImageFile=QString(myTopElement.firstChildElement("imageFile").text());
        return true;
    }
    else
    return false;
}
```

Here is the call graph for this function:



#### 6.3.3.3 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

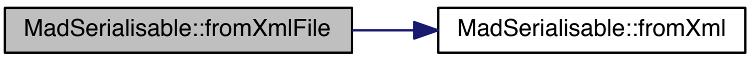
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



#### 6.3.3.4 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

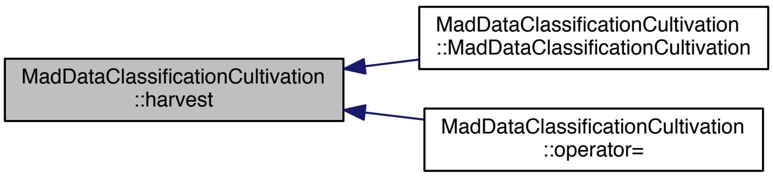
```
{
    return mGuid;
}
```

#### 6.3.3.5 MadSubCategory MadDataClassificationCultivation::harvest ( ) const

Definition at line 76 of file maddataclassificationcultivation.cpp.

```
{
    return mHarvest;
}
```

Here is the caller graph for this function:

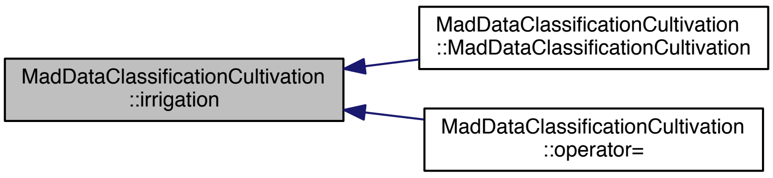

MadDataClassificationCultivation ::harvest
MadDataClassificationCultivation ::MadDataClassificationCultivation
MadDataClassificationCultivation ::operator=

#### 6.3.3.6 MadSubCategory MadDataClassificationCultivation::irrigation ( ) const

Definition at line 84 of file maddataclassificationcultivation.cpp.

```
{
    return mIrrigation;
}
```

Here is the caller graph for this function:

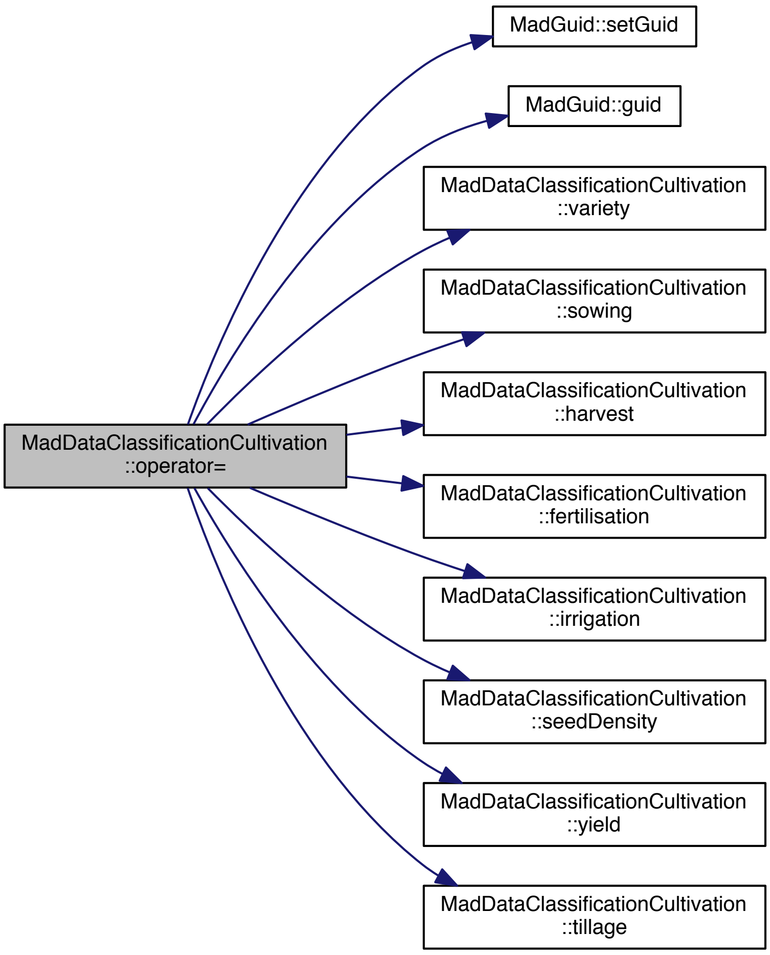

MadDataClassificationCultivation ::irrigation
MadDataClassificationCultivation ::MadDataClassificationCultivation
MadDataClassificationCultivation ::operator=

#### 6.3.3.7 MadDataClassificationCultivation & MadDataClassificationCultivation::operator= ( const MadDataClassificationCultivation & *theData* )

Definition at line 51 of file maddataclassificationcultivation.cpp.

```
{
    // gracefully handles self assignment
    if (this == &theData) return *this;
    setGuid(theData.guid());
    mVariety=theData.variety();
    mSowing=theData.sowing();
    mHarvest=theData.harvest();
    mFertilisation=theData.fertilisation();
    mIrrigation=theData.irrigation();
    mSeedDensity=theData.seedDensity();
    mYield=theData.yield();
    mTillage=theData.tillage();
    return *this;
}
```

Here is the call graph for this function:

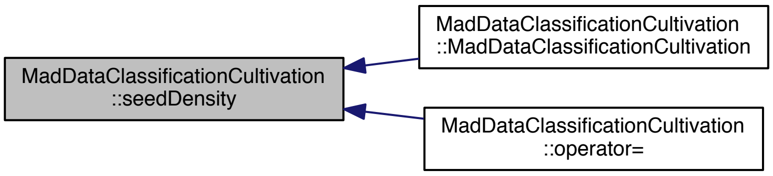

MadGuid::setGuid
MadGuid::guid
MadDataClassificationCultivation ::variety
MadDataClassificationCultivation ::sowing
MadDataClassificationCultivation ::harvest
MadDataClassificationCultivation ::operator=
MadDataClassificationCultivation ::fertilisation
MadDataClassificationCultivation ::irrigation
MadDataClassificationCultivation ::seedDensity
MadDataClassificationCultivation ::yield
MadDataClassificationCultivation ::tillage

#### 6.3.3.8 MadSubCategory MadDataClassificationCultivation::seedDensity ( ) const

Definition at line 88 of file maddataclassificationcultivation.cpp.

```
{
    return mSeedDensity;
}
```

Here is the caller graph for this function:

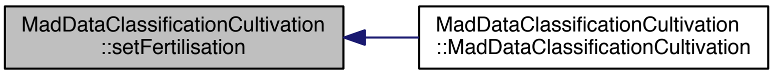

MadDataClassificationCultivation ::seedDensity
MadDataClassificationCultivation ::MadDataClassificationCultivation
MadDataClassificationCultivation ::operator=

#### 6.3.3.9 void MadDataClassificationCultivation::setFertilisation ( MadSubCategory *theData* )

Definition at line 114 of file maddataclassificationcultivation.cpp.

```
{
    mFertilisation = theData;
}
```

Here is the caller graph for this function:


MadDataClassificationCultivation ::setFertilisation
MadDataClassificationCultivation ::MadDataClassificationCultivation

#### 6.3.3.10 void MadGuid::setGuid ( QString *theGuid* = "" ) [inherited]

MadGuid::setGuid.

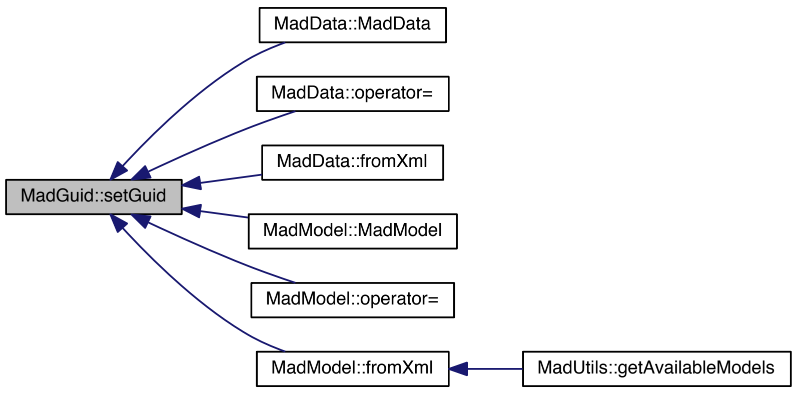
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:

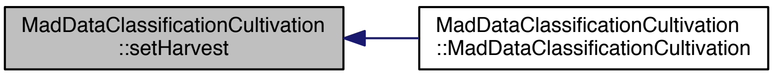


#### 6.3.3.11 void MadDataClassificationCultivation::setHarvest ( **MadSubCategory** *theData* )

Definition at line 110 of file maddataclassificationcultivation.cpp.

```
{
    mHarvest = theData;
}
```

Here is the caller graph for this function:

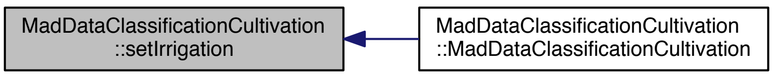

MadDataClassificationCultivation ::setHarvest
MadDataClassificationCultivation ::MadDataClassificationCultivation

#### 6.3.3.12 void MadDataClassificationCultivation::setIrrigation ( **MadSubCategory** *theData* )

Definition at line 118 of file maddataclassificationcultivation.cpp.

```
{
    mIrrigation = theData;
}
```

Here is the caller graph for this function:

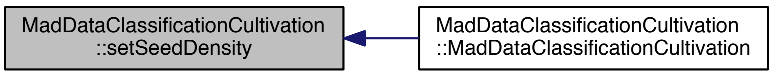

MadDataClassificationCultivation ::setIrrigation
MadDataClassificationCultivation ::MadDataClassificationCultivation

#### 6.3.3.13 void MadDataClassificationCultivation::setSeedDensity ( MadSubCategory *theData* )

Definition at line 122 of file maddataclassificationcultivation.cpp.

```
{
    mSeedDensity = theData;
}
```

Here is the caller graph for this function:

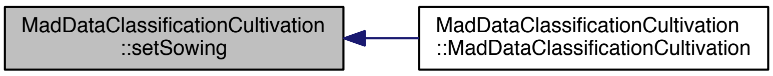

MadDataClassificationCultivation ::setSeedDensity
MadDataClassificationCultivation ::MadDataClassificationCultivation

#### 6.3.3.14 void MadDataClassificationCultivation::setSowing ( MadSubCategory *theData* )

Definition at line 106 of file maddataclassificationcultivation.cpp.

```
{
    mSowing = theData;
}
```

Here is the caller graph for this function:

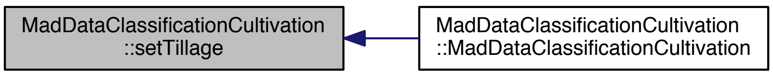

MadDataClassificationCultivation ::setSowing
MadDataClassificationCultivation ::MadDataClassificationCultivation

#### 6.3.3.15 void MadDataClassificationCultivation::setTillage ( MadSubCategory *theData* )

Definition at line 130 of file maddataclassificationcultivation.cpp.

```
{
    mTillage = theData;
}
```

Here is the caller graph for this function:

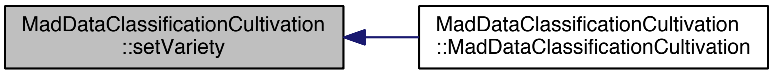

MadDataClassificationCultivation ::setTillage
MadDataClassificationCultivation ::MadDataClassificationCultivation

#### 6.3.3.16 void MadDataClassificationCultivation::setVariety ( **MadSubCategory** *theData* )

Definition at line 102 of file maddataclassificationcultivation.cpp.

```
{
    mVariety = theData;
}
```

Here is the caller graph for this function:

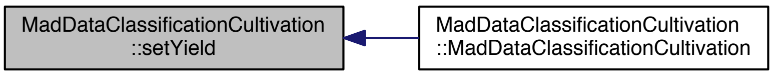

MadDataClassificationCultivation ::setVariety
MadDataClassificationCultivation ::MadDataClassificationCultivation

#### 6.3.3.17 void MadDataClassificationCultivation::setYield ( **MadSubCategory** *theData* )

Definition at line 126 of file maddataclassificationcultivation.cpp.

```
{
    mYield = theData;
}
```

Here is the caller graph for this function:

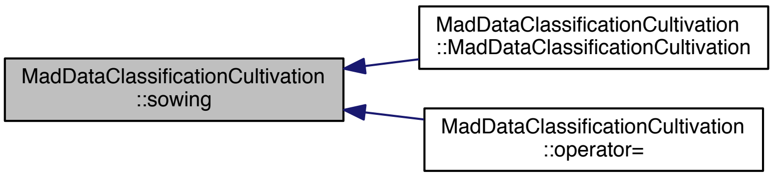

MadDataClassificationCultivation ::setYield
MadDataClassificationCultivation ::MadDataClassificationCultivation

#### 6.3.3.18 **MadSubCategory** MadDataClassificationCultivation::sowing ( ) const

Definition at line 72 of file maddataclassificationcultivation.cpp.

```
{
    return mSowing;
}
```

Here is the caller graph for this function:

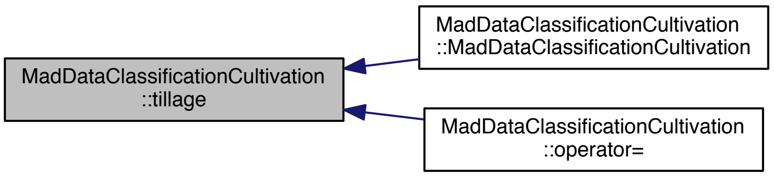

MadDataClassificationCultivation ::sowing
MadDataClassificationCultivation ::MadDataClassificationCultivation
MadDataClassificationCultivation ::operator=

#### 6.3.3.19 MadSubCategory MadDataClassificationCultivation::tillage ( ) const

Definition at line 96 of file maddataclassificationcultivation.cpp.

```
{
    return mTillage;
}
```

Here is the caller graph for this function:


MadDataClassificationCultivation ::tillage
MadDataClassificationCultivation ::MadDataClassificationCultivation
MadDataClassificationCultivation ::operator=

#### 6.3.3.20 QString MadDataClassificationCultivation::toHtml ( )

Return a html text representation of this layer

Definition at line 206 of file maddataclassificationcultivation.cpp.

```
{
  QString myString;
  //myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  //myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";

  //
  // the following shows example of how to do a couple of things
  //

  //myString+="<tr><td><b>Cals/Kg: </b></td><td>" + QString::number(mCropCalories) + "</td></tr>";
  //QString myCropFodderEnergyType = (mCropFodderEnergyType==0) ? "KCalories" : "TDN";
  //QString myUnits = (mAreaUnits==0) ? "Dunum" : "Hectare";
  //myString+="<tr><td><b>Fodder (kg/" + myUnits + "): </b></td><td>" +
   QString::number(mCropFodderProduction) + "</td></tr>";
  //myString+="<tr><td><b>Fodder Value/Kg: </b></td><td>" + QString::number(mCropFodderValue) +
   "</td></tr>";
  //myString+="<tr><td><b>FodderEnergyType: </b></td><td>" + myCropFodderEnergyType + "</td></tr>";
  //myString+="<tr><td><b>AreaUnits: </b></td><td>" + myUnits + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

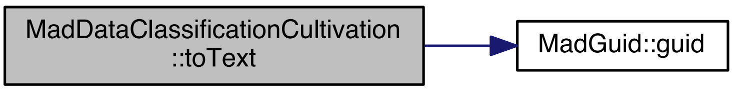
#### 6.3.3.21 QString MadDataClassificationCultivation::toText ( )

Return a plain text representation of this layer

Definition at line 197 of file maddataclassificationcultivation.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  //myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.3.3.22 QString MadDataClassificationCultivation::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

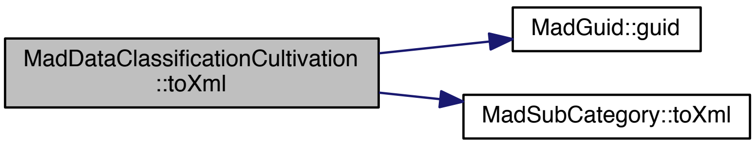
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

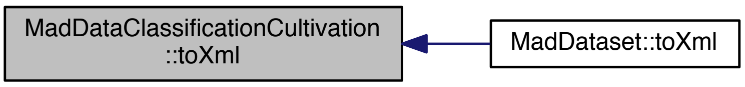
Definition at line 155 of file maddataclassificationcultivation.cpp.

```
{
  QString myString;
  myString+=QString("  <cultivation guid=\"" + guid() + "\">\n");

  myString+=QString("    <variety>\n");
  myString+=mVariety.toXml();
  myString+=QString("    </variety>\n");

  myString+=QString("    <sowing>\n");
  myString+=mSowing.toXml();
  myString+=QString("    </sowing>\n");

  myString+=QString("    <harvest>\n");
  myString+=mHarvest.toXml();
  myString+=QString("    </harvest>\n");

  myString+=QString("    <fertilisation>\n");
  myString+=mFertilisation.toXml();
  myString+=QString("    </fertilisation>\n");

  myString+=QString("    <irrigation>\n");
  myString+=mIrrigation.toXml();
  myString+=QString("    </irrigation>\n");

  myString+=QString("    <seeddensity>\n");
  myString+=mSeedDensity.toXml();
  myString+=QString("    </seeddensity>\n");

  myString+=QString("    <yield>\n");
  myString+=mYield.toXml();
  myString+=QString("    </yield>\n");

  myString+=QString("    <tillage>\n");
  myString+=mTillage.toXml();
  myString+=QString("    </tillage>\n");

  myString+=QString("  </cultivation>\n");

  return myString;
}
```

Here is the call graph for this function:



Here is the caller graph for this function:



#### 6.3.3.23 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

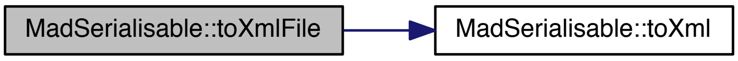
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:

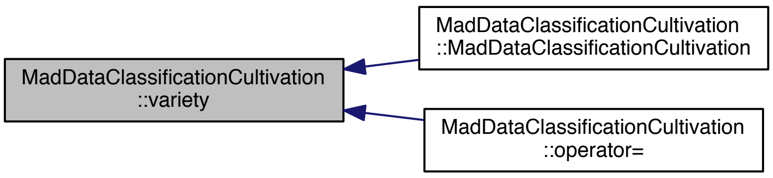


#### 6.3.3.24 MadSubCategory MadDataClassificationCultivation::variety ( ) const

Definition at line 68 of file maddataclassificationcultivation.cpp.

```
{
    return mVariety;
}
```

Here is the caller graph for this function:

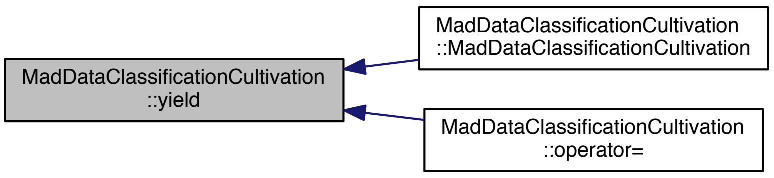

MadDataClassificationCultivation ::variety
MadDataClassificationCultivation ::MadDataClassificationCultivation
MadDataClassificationCultivation ::operator=

#### 6.3.3.25 MadSubCategory MadDataClassificationCultivation::yield ( ) const

Definition at line 92 of file maddataclassificationcultivation.cpp.

```
{
    return mYield;
}
```

Here is the caller graph for this function:


MadDataClassificationCultivation ::yield
MadDataClassificationCultivation ::MadDataClassificationCultivation
MadDataClassificationCultivation ::operator=

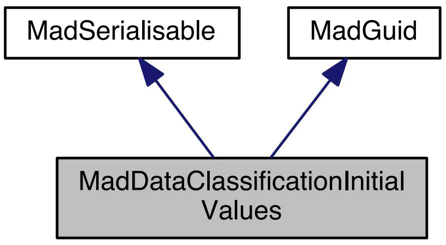
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationcultivation.-h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationcultivation.-cpp

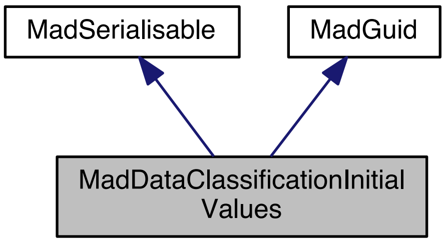
## 6.4 MadDataClassificationInitialValues Class Reference

```
#include <maddataclassificationinitialvalues.h>
```

Inheritance diagram for MadDataClassificationInitialValues:



Collaboration diagram for MadDataClassificationInitialValues:



### Public Member Functions

- MadDataClassificationInitialValues ()
- MadDataClassificationInitialValues (const MadDataClassificationInitialValues &theData)
- MadDataClassificationInitialValues & operator= (const MadDataClassificationInitialValues &theData)
- MadSubCategory soilMoisture () const
- MadSubCategory nitrogenMin () const
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- void setSoilMoisture (MadSubCategory theData)
- void setNitrogenMin (MadSubCategory theData)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

## 6.4.1 Detailed Description

Definition at line 36 of file maddataclassificationinitialvalues.h.

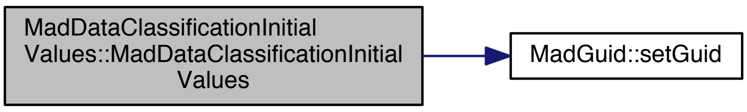
## 6.4.2 Constructor & Destructor Documentation

### 6.4.2.1 MadDataClassificationInitialValues::MadDataClassificationInitialValues ( )

Definition at line 34 of file maddataclassificationinitialvalues.cpp.

```
                                                                        :
     MadSerialisable(), MadGuid()
{
  setGuid();
}
```

Here is the call graph for this function:

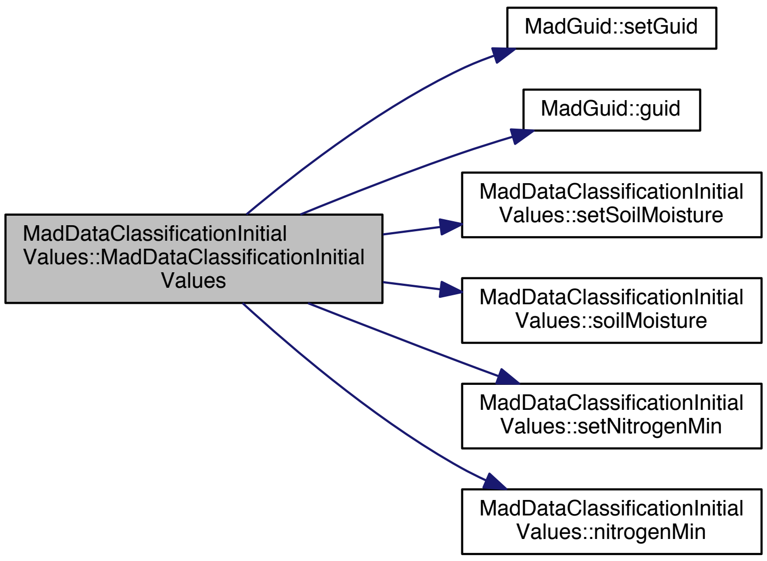

MadDataClassificationInitialValues::MadDataClassificationInitialValues
MadGuid::setGuid

#### 6.4.2.2 MadDataClassificationInitialValues::MadDataClassificationInitialValues ( const MadDataClassificationInitialValues & *theData* )

Definition at line 39 of file maddataclassificationinitialvalues.cpp.

```
{
  setGuid(theData.guid());
  setSoilMoisture(theData.soilMoisture());
  setNitrogenMin(theData.nitrogenMin());
}
```

Here is the call graph for this function:


MadDataClassificationInitialValues::MadDataClassificationInitialValues
MadGuid::setGuid
MadGuid::guid
MadDataClassificationInitialValues::setSoilMoisture
MadDataClassificationInitialValues::soilMoisture
MadDataClassificationInitialValues::setNitrogenMin
MadDataClassificationInitialValues::nitrogenMin

### 6.4.3 Member Function Documentation

#### 6.4.3.1 bool MadDataClassificationInitialValues::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

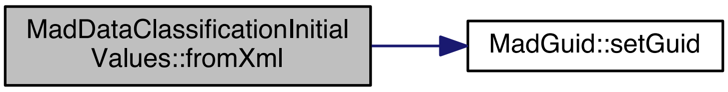
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 77 of file maddataclassificationinitialvalues.cpp.

```
{
    QDomDocument myDocument("mydocument");
    myDocument.setContent(theXml);
    QDomElement myTopElement = myDocument.firstChildElement("initialvalues");
    if (myTopElement.isNull())
    {
        //TODO - just make this a warning
        qDebug("the top element couldn't be found!");
        setGuid(myTopElement.attribute("guid"));
        //mName=MadUtils::xmlDecode(myTopElement.firstChildElement("name").text());
        //mDescription=MadUtils::xmlDecode(myTopElement.firstChildElement("description").text());
        //mImageFile=QString(myTopElement.firstChildElement("imageFile").text());
        return true;
    }
    else
    return false;
}
```

Here is the call graph for this function:



#### 6.4.3.2 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

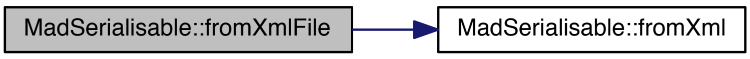
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



### 6.4.3.3 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

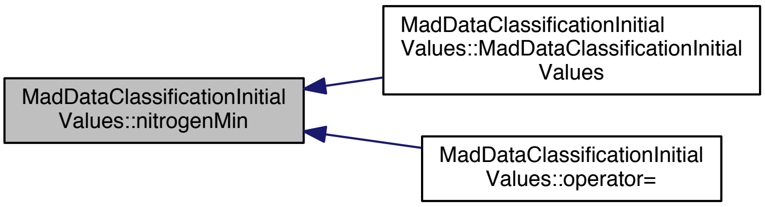
```
{
    return mGuid;
}
```

### 6.4.3.4 MadSubCategory MadDataClassificationInitialValues::nitrogenMin ( ) const

Definition at line 60 of file maddataclassificationinitialvalues.cpp.

```
{
  return mNitrogenMin;
}
```

Here is the caller graph for this function:

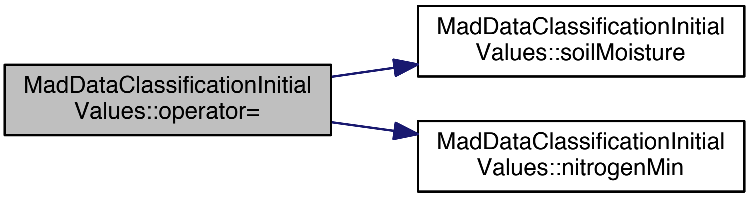

MadDataClassificationInitialValues::nitrogenMin
MadDataClassificationInitialValues::MadDataClassificationInitialValues
MadDataClassificationInitialValues::operator=

#### 6.4.3.5 MadDataClassificationInitialValues & MadDataClassificationInitialValues::operator= ( const MadDataClassificationInitialValues & *theData* )

Definition at line 46 of file maddataclassificationinitialvalues.cpp.

```
{
  // gracefully handles self assignment
  if (this == &theData) return *this;
  mSoilMoisture = theData.soilMoisture();
  mNitrogenMin = theData.nitrogenMin();
  return *this;
}
```

Here is the call graph for this function:


MadDataClassificationInitialValues::operator=
MadDataClassificationInitialValues::soilMoisture
MadDataClassificationInitialValues::nitrogenMin

#### 6.4.3.6 void MadGuid::setGuid ( QString *theGuid* = "" ) [inherited]

MadGuid::setGuid.

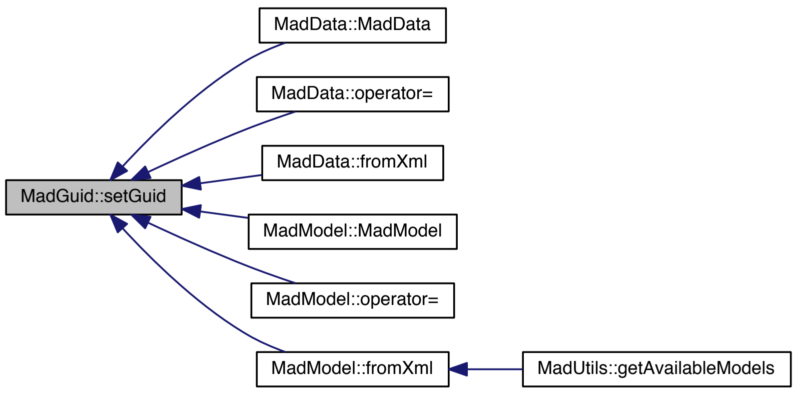
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:

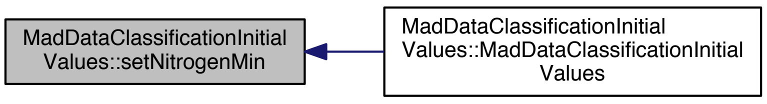


#### 6.4.3.7 void MadDataClassificationInitialValues::setNitrogenMin ( MadSubCategory *theData* )

Definition at line 71 of file maddataclassificationinitialvalues.cpp.

```
{
  mNitrogenMin = theData;
}
```

Here is the caller graph for this function:

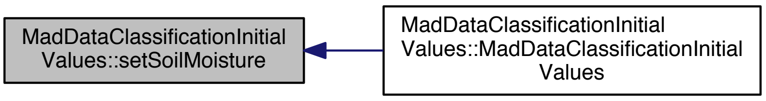

MadDataClassificationInitial Values::setNitrogenMin
MadDataClassificationInitial Values::MadDataClassificationInitial Values

#### 6.4.3.8 void MadDataClassificationInitialValues::setSoilMoisture ( MadSubCategory *theData* )

Definition at line 66 of file maddataclassificationinitialvalues.cpp.

```
{
  mSoilMoisture = theData;
}
```

Here is the caller graph for this function:

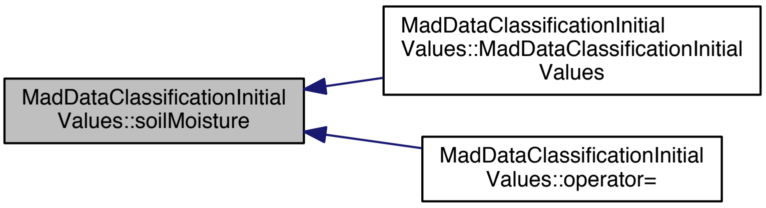

MadDataClassificationInitial Values::setSoilMoisture
MadDataClassificationInitial Values::MadDataClassificationInitial Values

#### 6.4.3.9 MadSubCategory MadDataClassificationInitialValues::soilMoisture ( ) const

Definition at line 56 of file maddataclassificationinitialvalues.cpp.

```
{
  return mSoilMoisture;
}
```

Here is the caller graph for this function:


MadDataClassificationInitial Values::soilMoisture
MadDataClassificationInitial Values::MadDataClassificationInitial Values
MadDataClassificationInitial Values::operator=

#### 6.4.3.10 QString MadDataClassificationInitialValues::toHtml ( )

Return a html text representation of this layer

Definition at line 123 of file maddataclassificationinitialvalues.cpp.

```
{
  QString myString;
  //myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  //myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";

  //
  // the following shows example of how to do a couple of things
  //

  //myString+="<tr><td><b>Cals/Kg: </b></td><td>" + QString::number(mCropCalories) + "</td></tr>";
  //QString myCropFodderEnergyType = (mCropFodderEnergyType==0) ? "KCalories" : "TDN";
  //QString myUnits = (mAreaUnits==0) ? "Dunum" : "Hectare";
  //myString+="<tr><td><b>Fodder (kg/" + myUnits + "): </b></td><td>" +
   QString::number(mCropFodderProduction) + "</td></tr>";
  //myString+="<tr><td><b>Fodder Value/Kg: </b></td><td>" + QString::number(mCropFodderValue) +
   "</td></tr>";
  //myString+="<tr><td><b>FodderEnergyType: </b></td><td>" + myCropFodderEnergyType + "</td></tr>";
  //myString+="<tr><td><b>AreaUnits: </b></td><td>" + myUnits + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

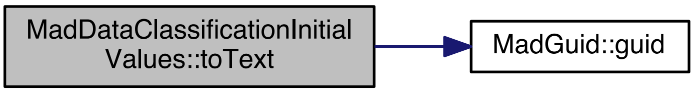
#### 6.4.3.11 QString MadDataClassificationInitialValues::toText ( )

Return a plain text representation of this layer

Definition at line 114 of file maddataclassificationinitialvalues.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  //myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.4.3.12 QString MadDataClassificationInitialValues::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

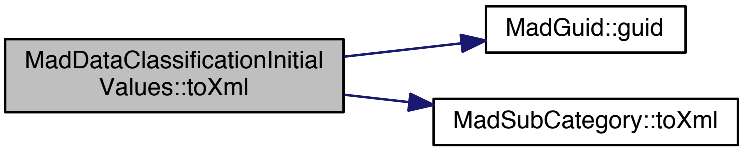
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

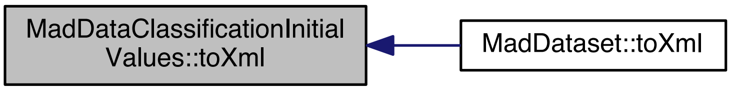
Definition at line 96 of file maddataclassificationinitialvalues.cpp.

```
{
 QString myString;
 myString+=QString("  <initialvalues guid=\"" + guid() + "\">\n");

  myString+=QString("    <soilmoisture>\n");
  myString+=mSoilMoisture.toXml();
  myString+=QString("    </soilmoisture>\n");

  myString+=QString("    <nmin>\n");
  myString+=mNitrogenMin.toXml();
  myString+=QString("    </nmin>\n");

  myString+=QString("  </initialvalues>\n");
  return myString;

}
```

Here is the call graph for this function:



Here is the caller graph for this function:



#### 6.4.3.13 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

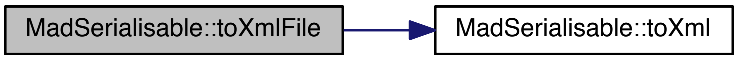
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
        //@TODO Error handler!
        myResult=false;
    }
    return myResult ;
}
```

Here is the call graph for this function:



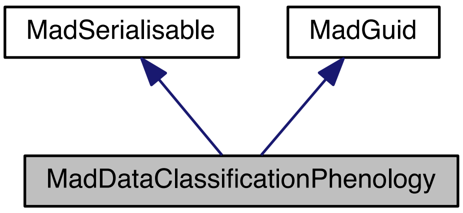
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationinitialvalues.-h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationinitialvalues.-cpp

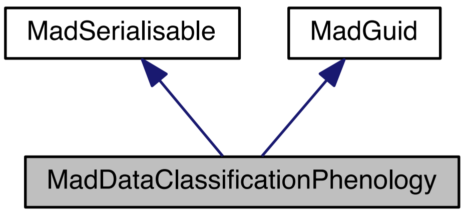
## 6.5 MadDataClassificationPhenology Class Reference

`#include <maddataclassificationphenology.h>`

Inheritance diagram for MadDataClassificationPhenology:



Collaboration diagram for MadDataClassificationPhenology:



## Public Member Functions

- MadDataClassificationPhenology ()
- MadDataClassificationPhenology (const MadDataClassificationPhenology &theData)
- MadDataClassificationPhenology & operator= (const MadDataClassificationPhenology &theData)
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- MadSubCategory emergence () const
- MadSubCategory stemElongation () const
- MadSubCategory earEmergence () const
- MadSubCategory flowering () const
- MadSubCategory yellowRipeness () const
- void setEmergence (MadSubCategory theData)
- void setStemElongation (MadSubCategory theData)
- void setEarEmergence (MadSubCategory theData)
- void setFlowering (MadSubCategory theData)
- void setYellowRipeness (MadSubCategory theData)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.5.1 Detailed Description

Definition at line 36 of file maddataclassificationphenology.h.

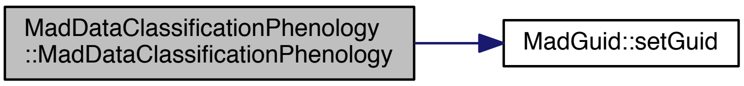
### 6.5.2 Constructor & Destructor Documentation

#### 6.5.2.1 MadDataClassificationPhenology::MadDataClassificationPhenology ( )

Definition at line 33 of file maddataclassificationphenology.cpp.

```
                                                                   : MadSerialisable(),
     MadGuid()
{
  setGuid();
}
```

Here is the call graph for this function:

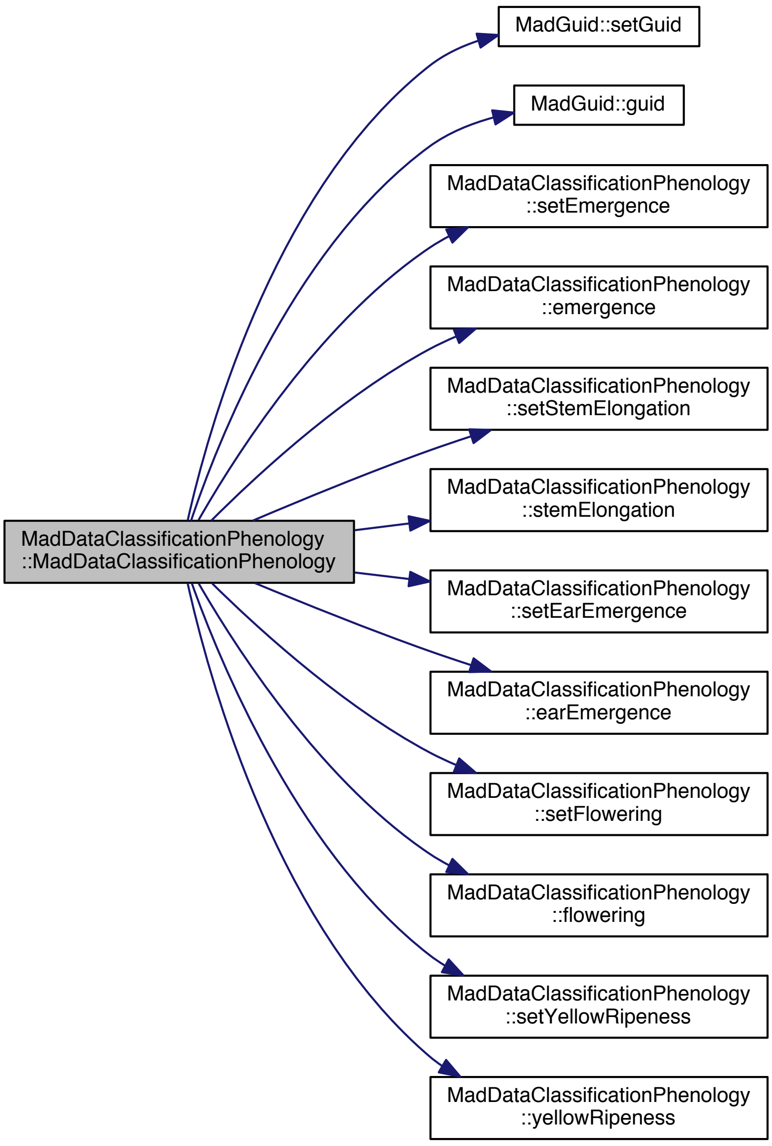

MadDataClassificationPhenology ::MadDataClassificationPhenology
MadGuid::setGuid

#### 6.5.2.2 MadDataClassificationPhenology::MadDataClassificationPhenology ( const MadDataClassificationPhenology & *theData* )

Definition at line 38 of file maddataclassificationphenology.cpp.

```
{
  setGuid(theData.guid());
  setEmergence(theData.emergence());
  setStemElongation(theData.stemElongation());
  setEarEmergence(theData.earEmergence());
  setFlowering(theData.flowering());
  setYellowRipeness(theData.yellowRipeness());
}
```

Here is the call graph for this function:


MadGuid::setGuid
MadGuid::guid
MadDataClassificationPhenology ::setEmergence
MadDataClassificationPhenology ::emergence
MadDataClassificationPhenology ::setStemElongation
MadDataClassificationPhenology ::stemElongation
MadDataClassificationPhenology ::MadDataClassificationPhenology
MadDataClassificationPhenology ::setEarEmergence
MadDataClassificationPhenology ::earEmergence
MadDataClassificationPhenology ::setFlowering
MadDataClassificationPhenology ::flowering
MadDataClassificationPhenology ::setYellowRipeness
MadDataClassificationPhenology ::yellowRipeness

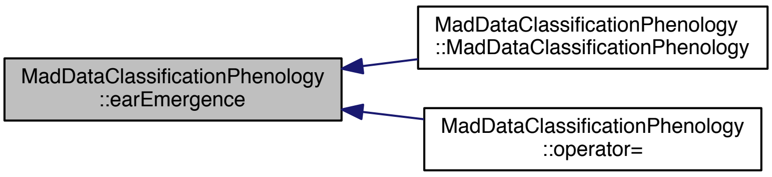
### 6.5.3 Member Function Documentation

#### 6.5.3.1 MadSubCategory MadDataClassificationPhenology::earEmergence ( ) const

Definition at line 69 of file maddataclassificationphenology.cpp.

```
{
  return mEarEmergence;
}
```

Here is the caller graph for this function:

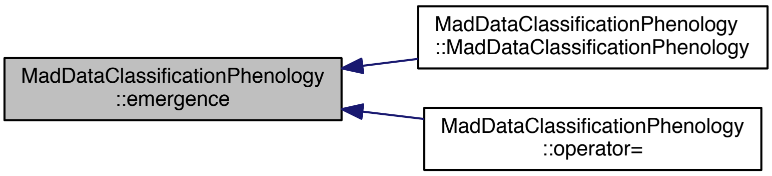

MadDataClassificationPhenology ::earEmergence
MadDataClassificationPhenology ::MadDataClassificationPhenology
MadDataClassificationPhenology ::operator=

#### 6.5.3.2 MadSubCategory MadDataClassificationPhenology::emergence ( ) const

Definition at line 61 of file maddataclassificationphenology.cpp.

```
{
  return mEmergence;
}
```

Here is the caller graph for this function:

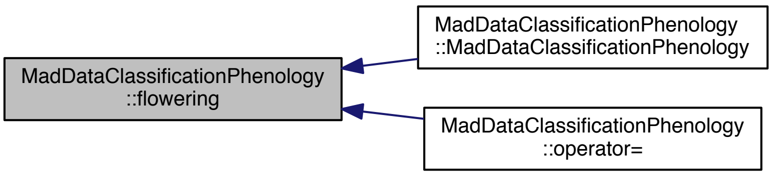

MadDataClassificationPhenology ::emergence
MadDataClassificationPhenology ::MadDataClassificationPhenology
MadDataClassificationPhenology ::operator=

#### 6.5.3.3 MadSubCategory MadDataClassificationPhenology::flowering ( ) const

Definition at line 73 of file maddataclassificationphenology.cpp.

```
{
  return mFlowering;
}
```

Here is the caller graph for this function:


MadDataClassificationPhenology ::flowering
MadDataClassificationPhenology ::MadDataClassificationPhenology
MadDataClassificationPhenology ::operator=

#### 6.5.3.4 bool MadDataClassificationPhenology::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

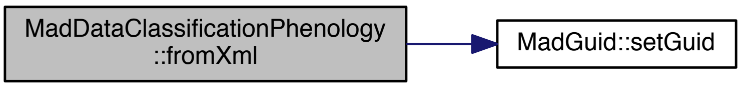
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 109 of file maddataclassificationphenology.cpp.

```
{
    QDomDocument myDocument("mydocument");
    myDocument.setContent(theXml);
    QDomElement myTopElement = myDocument.firstChildElement("phenology");
    if (myTopElement.isNull())
    {
        //TODO - just make this a warning
        qDebug("the top element couldn't be found!");
        setGuid(myTopElement.attribute("guid"));
        //mName=MadUtils::xmlDecode(myTopElement.firstChildElement("name").text());
        //mDescription=MadUtils::xmlDecode(myTopElement.firstChildElement("description").text());
        //mImageFile=QString(myTopElement.firstChildElement("imageFile").text());
        return true;
    }
    else
    return false;
}
```

Here is the call graph for this function:



#### 6.5.3.5 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

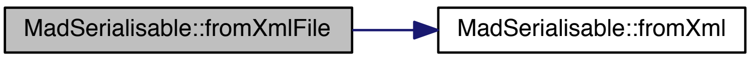
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



#### 6.5.3.6 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

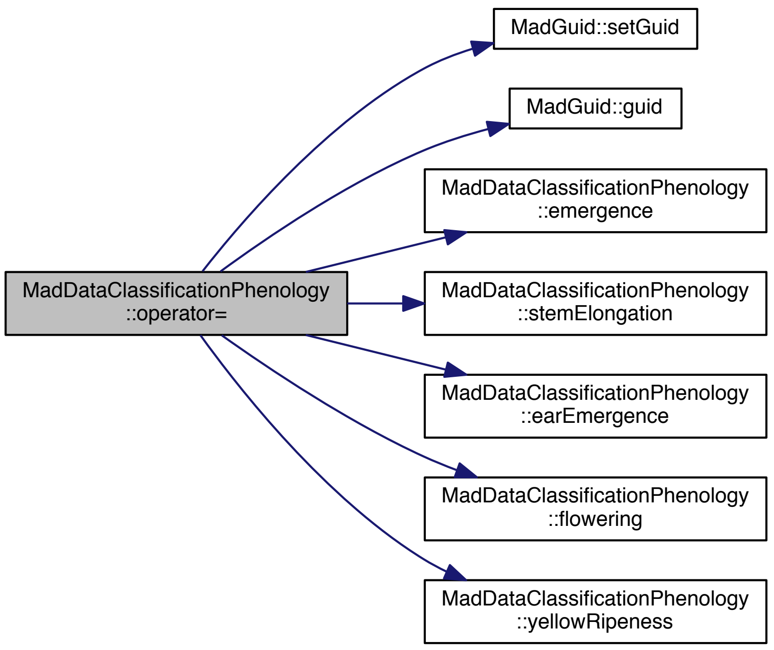
```
{
    return mGuid;
}
```

#### 6.5.3.7 MadDataClassificationPhenology & MadDataClassificationPhenology::operator= ( const MadDataClassificationPhenology & *theData* )

Definition at line 48 of file maddataclassificationphenology.cpp.

```
{
  // gracefully handles self assignment
  if (this == &theData) return *this;
  setGuid(theData.guid());
  mEmergence=theData.emergence();
  mStemElongation=theData.stemElongation();
  mEarEmergence=theData.earEmergence();
  mFlowering=theData.flowering();
  mYellowRipeness=theData.yellowRipeness();
  return *this;
}
```

Here is the call graph for this function:

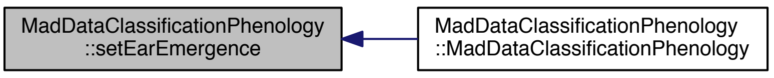

MadGuid::setGuid
MadGuid::guid
MadDataClassificationPhenology ::emergence
MadDataClassificationPhenology ::operator=
MadDataClassificationPhenology ::stemElongation
MadDataClassificationPhenology ::earEmergence
MadDataClassificationPhenology ::flowering
MadDataClassificationPhenology ::yellowRipeness

#### 6.5.3.8 void MadDataClassificationPhenology::setEarEmergence ( MadSubCategory *theData* )

Definition at line 93 of file maddataclassificationphenology.cpp.

```
{
  mEarEmergence = theData;
}
```

Here is the caller graph for this function:

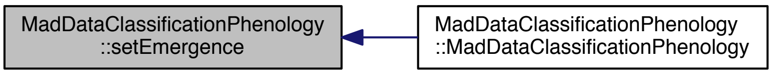

MadDataClassificationPhenology ::setEarEmergence
MadDataClassificationPhenology ::MadDataClassificationPhenology

#### 6.5.3.9 void MadDataClassificationPhenology::setEmergence ( MadSubCategory *theData* )

Definition at line 83 of file maddataclassificationphenology.cpp.

```
{
  mEmergence = theData;
}
```

Here is the caller graph for this function:

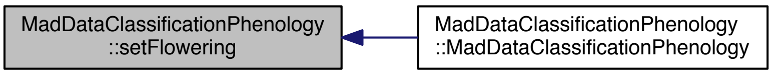

MadDataClassificationPhenology ::setEmergence
MadDataClassificationPhenology ::MadDataClassificationPhenology

#### 6.5.3.10 void MadDataClassificationPhenology::setFlowering ( MadSubCategory *theData* )

Definition at line 98 of file maddataclassificationphenology.cpp.

```
{
  mFlowering = theData;
}
```

Here is the caller graph for this function:


MadDataClassificationPhenology ::setFlowering
MadDataClassificationPhenology ::MadDataClassificationPhenology

#### 6.5.3.11 void MadGuid::setGuid ( QString *theGuid* = "" ) [inherited]

MadGuid::setGuid.

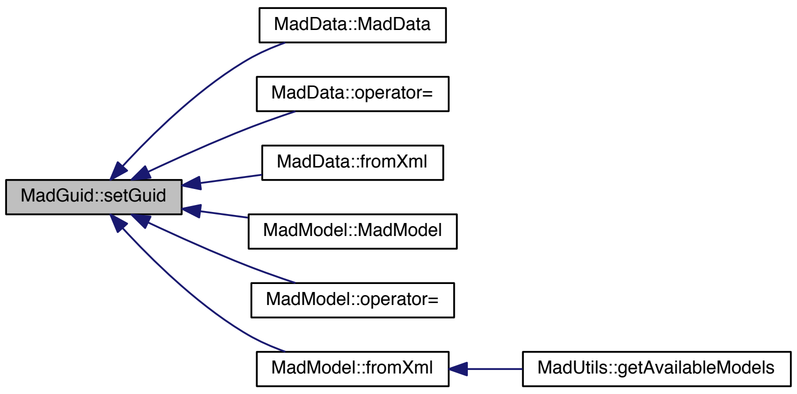
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:

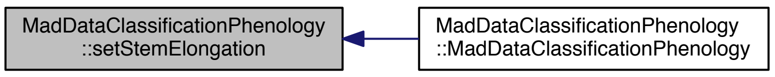


#### 6.5.3.12 void MadDataClassificationPhenology::setStemElongation ( **MadSubCategory** ***theData*** )

Definition at line 88 of file maddataclassificationphenology.cpp.

```
89 {
90   mStemElongation = theData;
91 }
```

Here is the caller graph for this function:

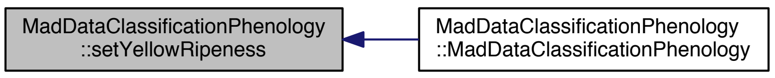

MadDataClassificationPhenology ::setStemElongation
MadDataClassificationPhenology ::MadDataClassificationPhenology

#### 6.5.3.13 void MadDataClassificationPhenology::setYellowRipeness ( **MadSubCategory** ***theData*** )

Definition at line 103 of file maddataclassificationphenology.cpp.

```
104 {
105   mYellowRipeness = theData;
106 }
```

Here is the caller graph for this function:

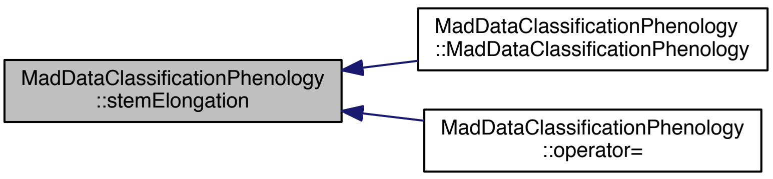

MadDataClassificationPhenology ::setYellowRipeness
MadDataClassificationPhenology ::MadDataClassificationPhenology

#### 6.5.3.14 MadSubCategory MadDataClassificationPhenology::stemElongation ( ) const

Definition at line 65 of file maddataclassificationphenology.cpp.

```
{
  return mStemElongation;
}
```

Here is the caller graph for this function:


MadDataClassificationPhenology ::stemElongation
MadDataClassificationPhenology ::MadDataClassificationPhenology
MadDataClassificationPhenology ::operator=

#### 6.5.3.15 QString MadDataClassificationPhenology::toHtml ( )

Return a html text representation of this layer

Definition at line 166 of file maddataclassificationphenology.cpp.

```
{
  QString myString;
  //myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  //myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";

  //
  // the following shows example of how to do a couple of things
  //

  //myString+="<tr><td><b>Cals/Kg: </b></td><td>" + QString::number(mCropCalories) + "</td></tr>";
  //QString myCropFodderEnergyType = (mCropFodderEnergyType==0) ? "KCalories" : "TDN";
  //QString myUnits = (mAreaUnits==0) ? "Dunum" : "Hectare";
  //myString+="<tr><td><b>Fodder (kg/" + myUnits + "): </b></td><td>" +
   QString::number(mCropFodderProduction) + "</td></tr>";
  //myString+="<tr><td><b>Fodder Value/Kg: </b></td><td>" + QString::number(mCropFodderValue) +
   "</td></tr>";
  //myString+="<tr><td><b>FodderEnergyType: </b></td><td>" + myCropFodderEnergyType + "</td></tr>";
  //myString+="<tr><td><b>AreaUnits: </b></td><td>" + myUnits + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

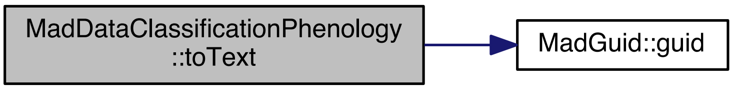
#### 6.5.3.16 QString MadDataClassificationPhenology::toText ( )

Return a plain text representation of this layer

Definition at line 157 of file maddataclassificationphenology.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  //myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.5.3.17 QString MadDataClassificationPhenology::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

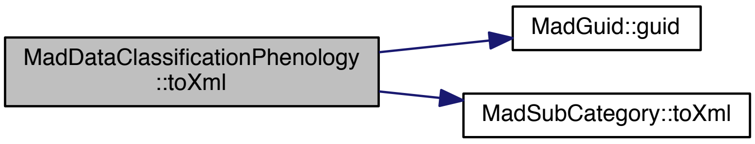
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

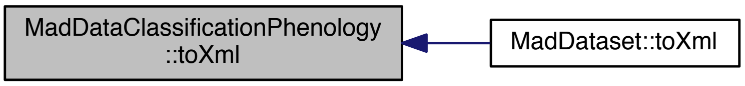
Definition at line 128 of file maddataclassificationphenology.cpp.

```
{
  QString myString;
  myString+=QString("  <phenology guid=\"" + guid() + "\">\n");

  myString+=QString("    <emergence>\n");
  myString+=mEmergence.toXml();
  myString+=QString("    </emergence>\n");

  myString+=QString("    <stemelongation>\n");
  myString+=mStemElongation.toXml();
  myString+=QString("    </stemelongation>\n");

  myString+=QString("    <earemergence>\n");
  myString+=mEarEmergence.toXml();
  myString+=QString("    </earemergence>\n");

  myString+=QString("    <flowering>\n");
  myString+=mFlowering.toXml();
  myString+=QString("    </flowering>\n");

  myString+=QString("    <yellowripeness>\n");
  myString+=mYellowRipeness.toXml();
  myString+=QString("    </yellowripeness>\n");

  myString+=QString("  </phenology>\n");
  return myString;
}
```

Here is the call graph for this function:



Here is the caller graph for this function:



#### 6.5.3.18 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

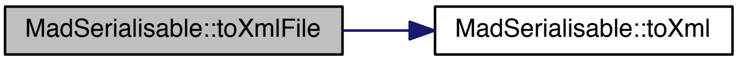
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:

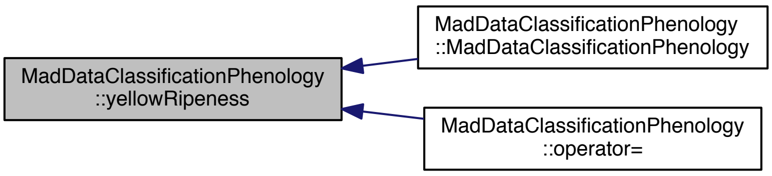


#### 6.5.3.19 MadSubCategory MadDataClassificationPhenology::yellowRipeness ( ) const

Definition at line 77 of file maddataclassificationphenology.cpp.

```
{
  return mYellowRipeness;
}
```

Here is the caller graph for this function:


MadDataClassificationPhenology ::yellowRipeness
MadDataClassificationPhenology ::MadDataClassificationPhenology
MadDataClassificationPhenology ::operator=

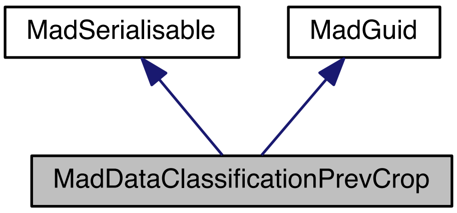
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationphenology.-h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationphenology.-cpp

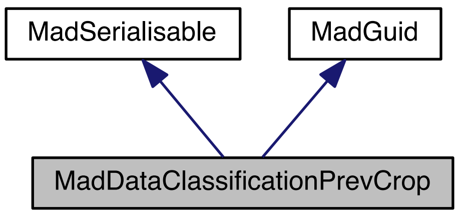
## 6.6 MadDataClassificationPrevCrop Class Reference

`#include <maddataclassificationprevcrop.h>`

Inheritance diagram for MadDataClassificationPrevCrop:



Collaboration diagram for MadDataClassificationPrevCrop:



### Public Member Functions

- MadDataClassificationPrevCrop ()
- MadDataClassificationPrevCrop (const MadDataClassificationPrevCrop &theData)
- MadDataClassificationPrevCrop & operator= (const MadDataClassificationPrevCrop &theData)
- MadSubCategory crop () const
- MadSubCategory sowingDate () const
- MadSubCategory harvestDate () const
- MadSubCategory yield () const
- MadSubCategory residueMgmt () const
- MadSubCategory fertilisation () const
- MadSubCategory irrigation () const
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- void setCrop (MadSubCategory theData)
- void setSowingDate (MadSubCategory theData)
- void setHarvestDate (MadSubCategory theData)
- void setYield (MadSubCategory theData)
- void setResidueMgmt (MadSubCategory theData)
- void setFertilisation (MadSubCategory theData)
- void setIrrigation (MadSubCategory theData)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.6.1 Detailed Description

Definition at line 35 of file maddataclassificationprevcrop.h.

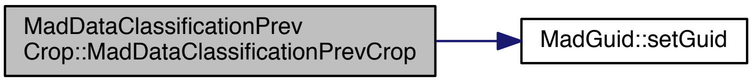
## 6.6.2 Constructor & Destructor Documentation

### 6.6.2.1 MadDataClassificationPrevCrop::MadDataClassificationPrevCrop ( )

Definition at line 33 of file maddataclassificationprevcrop.cpp.

```
                                                              : MadSerialisable(),
     MadGuid()
{
  setGuid();
}
```

Here is the call graph for this function:

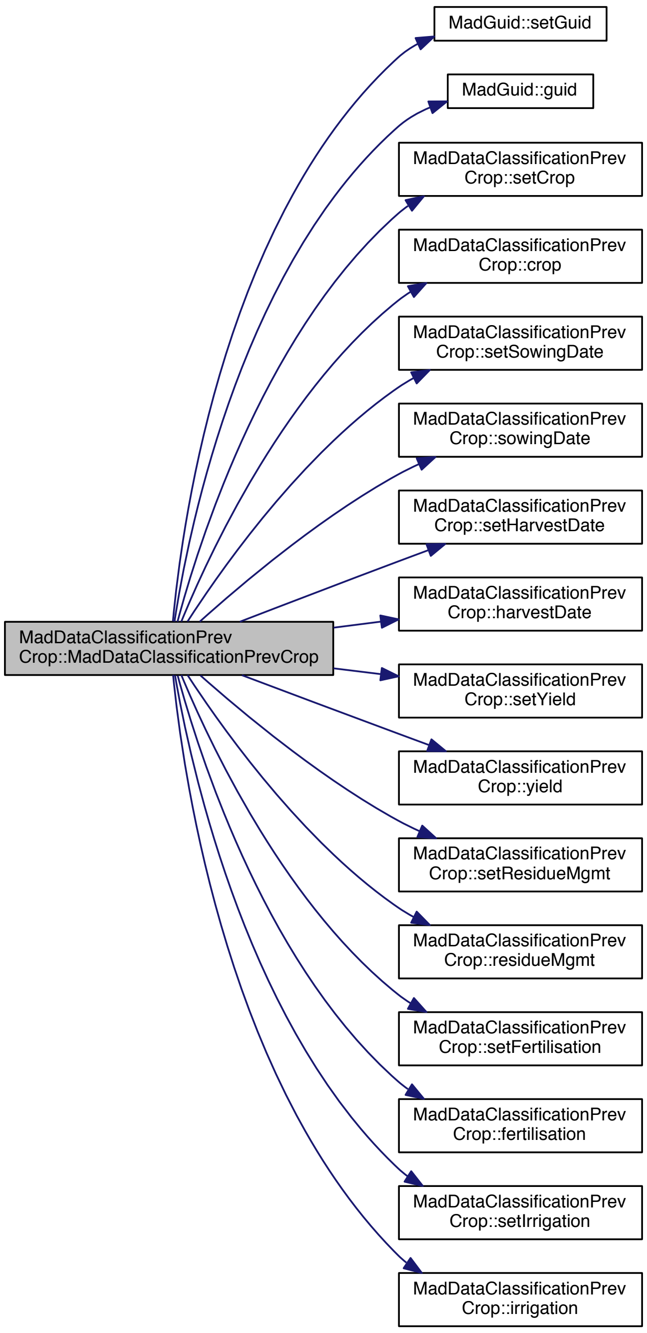

MadDataClassificationPrev Crop::MadDataClassificationPrevCrop
MadGuid::setGuid

### 6.6.2.2 MadDataClassificationPrevCrop::MadDataClassificationPrevCrop ( const MadDataClassificationPrevCrop & *theData* )

Definition at line 38 of file maddataclassificationprevcrop.cpp.

```
{
  setGuid(theData.guid());
  setCrop(theData.crop());
  setSowingDate(theData.sowingDate());
  setHarvestDate(theData.harvestDate());
  setYield(theData.yield());
  setResidueMgmt(theData.residueMgmt());
  setFertilisation(theData.fertilisation());
  setIrrigation(theData.irrigation());
}
```

Here is the call graph for this function:


MadGuid::setGuid
MadGuid::guid
MadDataClassificationPrev Crop::setCrop
MadDataClassificationPrev Crop::crop
MadDataClassificationPrev Crop::setSowingDate
MadDataClassificationPrev Crop::sowingDate
MadDataClassificationPrev Crop::setHarvestDate
MadDataClassificationPrev Crop::harvestDate
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop
MadDataClassificationPrev Crop::setYield
MadDataClassificationPrev Crop::yield
MadDataClassificationPrev Crop::setResidueMgmt
MadDataClassificationPrev Crop::residueMgmt
MadDataClassificationPrev Crop::setFertilisation
MadDataClassificationPrev Crop::fertilisation
MadDataClassificationPrev Crop::setIrrigation
MadDataClassificationPrev Crop::irrigation

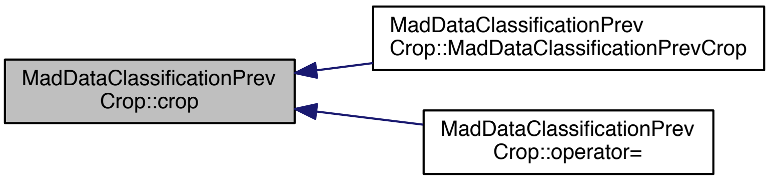
### 6.6.3 Member Function Documentation

#### 6.6.3.1 MadSubCategory MadDataClassificationPrevCrop::crop ( ) const

Definition at line 67 of file maddataclassificationprevcrop.cpp.

```
{
  return mCrop;
}
```

Here is the caller graph for this function:

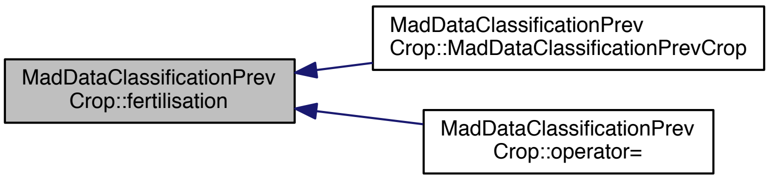

MadDataClassificationPrev Crop::crop
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop
MadDataClassificationPrev Crop::operator=

#### 6.6.3.2 MadSubCategory MadDataClassificationPrevCrop::fertilisation ( ) const

Definition at line 87 of file maddataclassificationprevcrop.cpp.

```
{
  return mFertilisation;
}
```

Here is the caller graph for this function:


MadDataClassificationPrev Crop::fertilisation
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop
MadDataClassificationPrev Crop::operator=

#### 6.6.3.3 bool MadDataClassificationPrevCrop::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

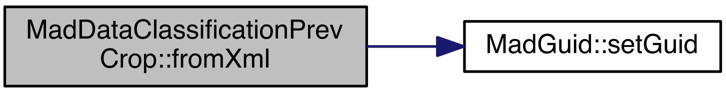
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 132 of file maddataclassificationprevcrop.cpp.

```
{
    QDomDocument myDocument("mydocument");
    myDocument.setContent(theXml);
    QDomElement myTopElement = myDocument.firstChildElement("prevcrop");
    if (myTopElement.isNull())
    {
        //TODO - just make this a warning
        qDebug("the top element couldn’t be found!");
        setGuid(myTopElement.attribute("guid"));
        //mName=MadUtils::xmlDecode(myTopElement.firstChildElement("name").text());
        //mDescription=MadUtils::xmlDecode(myTopElement.firstChildElement("description").text());
        //mImageFile=QString(myTopElement.firstChildElement("imageFile").text());
        return true;
    }
    else
    return false;
}
```

Here is the call graph for this function:



#### 6.6.3.4 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

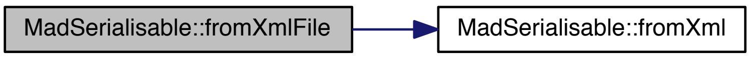
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



#### 6.6.3.5 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

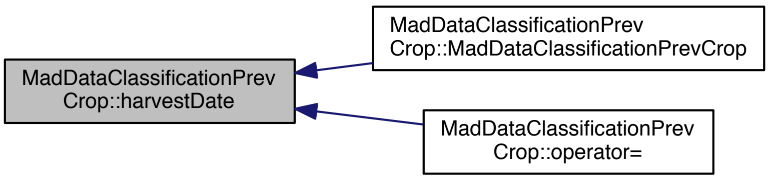
```
{
    return mGuid;
}
```

#### 6.6.3.6 MadSubCategory MadDataClassificationPrevCrop::harvestDate ( ) const

Definition at line 75 of file maddataclassificationprevcrop.cpp.

```
{
  return mHarvestDate;
}
```

Here is the caller graph for this function:

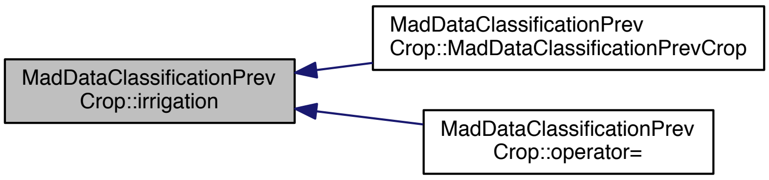

MadDataClassificationPrev Crop::harvestDate
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop
MadDataClassificationPrev Crop::operator=

#### 6.6.3.7 MadSubCategory MadDataClassificationPrevCrop::irrigation ( ) const

Definition at line 91 of file maddataclassificationprevcrop.cpp.

```
{
  return mIrrigation;
}
```

Here is the caller graph for this function:

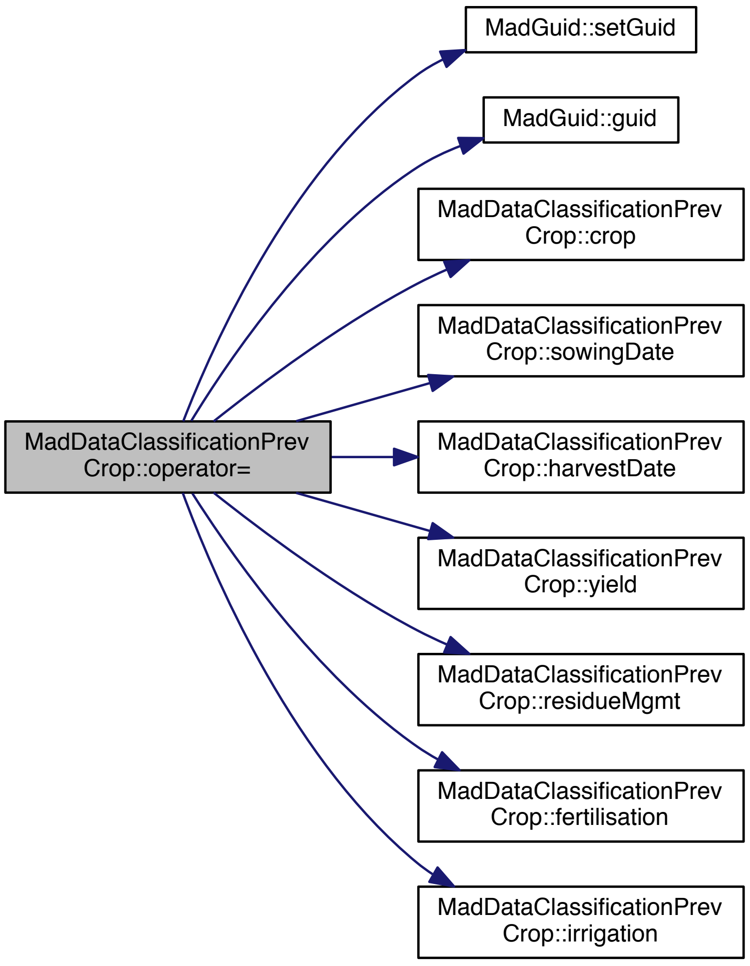

MadDataClassificationPrev Crop::irrigation
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop
MadDataClassificationPrev Crop::operator=

#### 6.6.3.8 MadDataClassificationPrevCrop & MadDataClassificationPrevCrop::operator= ( const MadDataClassificationPrevCrop & *theData* )

Definition at line 50 of file maddataclassificationprevcrop.cpp.

```
{
  // gracefully handles self assignment
  if (this == &theData) return *this;
  setGuid(theData.guid());
  mCrop=theData.crop();
  mSowingDate=theData.sowingDate();
  mHarvestDate=theData.harvestDate();
  mYield=theData.yield();
  mResidueMgmt=theData.residueMgmt();
  mFertilisation=theData.fertilisation();
  mIrrigation=theData.irrigation();
  return *this;
}
```

Here is the call graph for this function:

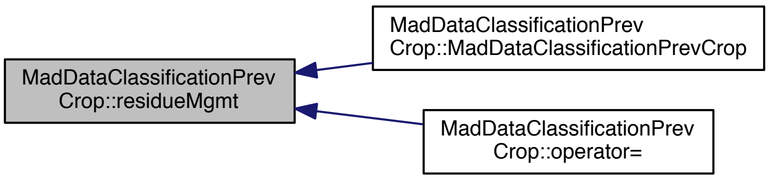

MadGuid::setGuid
MadGuid::guid
MadDataClassificationPrevCrop::crop
MadDataClassificationPrevCrop::sowingDate
MadDataClassificationPrevCrop::operator=
MadDataClassificationPrevCrop::harvestDate
MadDataClassificationPrevCrop::yield
MadDataClassificationPrevCrop::residueMgmt
MadDataClassificationPrevCrop::fertilisation
MadDataClassificationPrevCrop::irrigation

### 6.6.3.9 MadSubCategory MadDataClassificationPrevCrop::residueMgmt ( ) const

Definition at line 83 of file maddataclassificationprevcrop.cpp.

```
{
  return mResidueMgmt;
}
```

Here is the caller graph for this function:

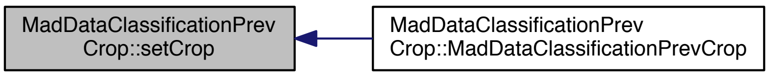

MadDataClassificationPrev Crop::residueMgmt
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop
MadDataClassificationPrev Crop::operator=

#### 6.6.3.10 void MadDataClassificationPrevCrop::setCrop ( MadSubCategory *theData* )

Definition at line 97 of file maddataclassificationprevcrop.cpp.

```
98 {
99   mCrop = theData;
100 }
```

Here is the caller graph for this function:

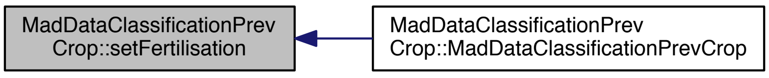

MadDataClassificationPrev Crop::setCrop
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop

#### 6.6.3.11 void MadDataClassificationPrevCrop::setFertilisation ( MadSubCategory *theData* )

Definition at line 122 of file maddataclassificationprevcrop.cpp.

```
123 {
124   mFertilisation = theData;
125 }
```

Here is the caller graph for this function:


MadDataClassificationPrev Crop::setFertilisation
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop

#### 6.6.3.12 void MadGuid::setGuid ( QString *theGuid* = "" ) [inherited]

MadGuid::setGuid.

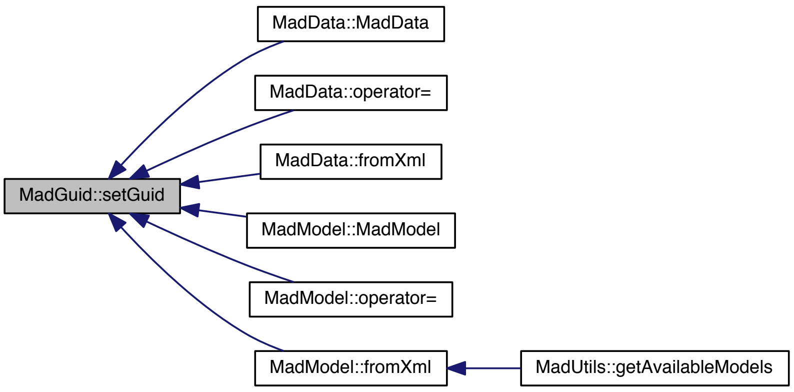
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:

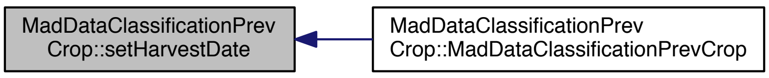


#### 6.6.3.13 void MadDataClassificationPrevCrop::setHarvestDate ( MadSubCategory *theData* )

Definition at line 107 of file maddataclassificationprevcrop.cpp.

```
{
  mHarvestDate = theData;
}
```

Here is the caller graph for this function:

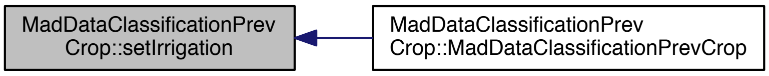

MadDataClassificationPrev Crop::setHarvestDate
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop

#### 6.6.3.14 void MadDataClassificationPrevCrop::setIrrigation ( MadSubCategory *theData* )

Definition at line 127 of file maddataclassificationprevcrop.cpp.

```
{
  mIrrigation = theData;
}
```

Here is the caller graph for this function:

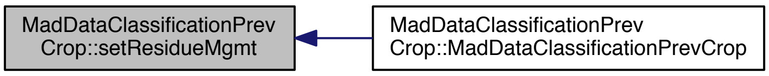

MadDataClassificationPrev Crop::setIrrigation
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop

#### 6.6.3.15 void MadDataClassificationPrevCrop::setResidueMgmt ( MadSubCategory *theData* )

Definition at line 117 of file maddataclassificationprevcrop.cpp.

```
{
  mResidueMgmt = theData;
}
```

Here is the caller graph for this function:

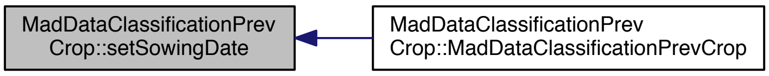

MadDataClassificationPrev Crop::setResidueMgmt
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop

#### 6.6.3.16 void MadDataClassificationPrevCrop::setSowingDate ( MadSubCategory *theData* )

Definition at line 102 of file maddataclassificationprevcrop.cpp.

```
{
  mSowingDate = theData;
}
```

Here is the caller graph for this function:

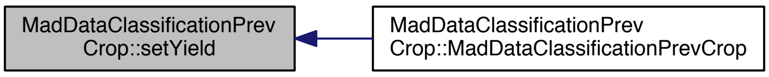

MadDataClassificationPrev Crop::setSowingDate
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop

#### 6.6.3.17 void MadDataClassificationPrevCrop::setYield ( MadSubCategory *theData* )

Definition at line 112 of file maddataclassificationprevcrop.cpp.

```
{
  mYield = theData;
}
```

Here is the caller graph for this function:

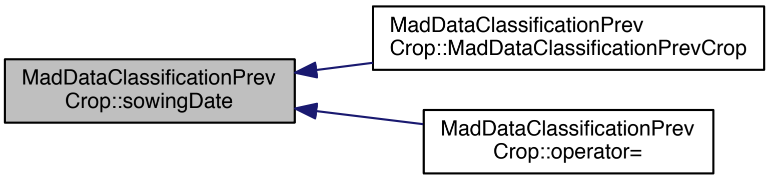

MadDataClassificationPrev Crop::setYield
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop

#### 6.6.3.18 MadSubCategory MadDataClassificationPrevCrop::sowingDate ( ) const

Definition at line 71 of file maddataclassificationprevcrop.cpp.

```
{
  return mSowingDate;
}
```

Here is the caller graph for this function:


MadDataClassificationPrev Crop::sowingDate
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop
MadDataClassificationPrev Crop::operator=

#### 6.6.3.19 QString MadDataClassificationPrevCrop::toHtml ( )

Return a html text representation of this layer

Definition at line 198 of file maddataclassificationprevcrop.cpp.

```
{
  QString myString;
  //myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  //myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";

  //
  // the following shows example of how to do a couple of things
  //

  //myString+="<tr><td><b>Cals/Kg: </b></td><td>" + QString::number(mCropCalories) + "</td></tr>";
  //QString myCropFodderEnergyType = (mCropFodderEnergyType==0) ? "KCalories" : "TDN";
  //QString myUnits = (mAreaUnits==0) ? "Dunum" : "Hectare";
  //myString+="<tr><td><b>Fodder (kg/" + myUnits + "): </b></td><td>" +
   QString::number(mCropFodderProduction) + "</td></tr>";
  //myString+="<tr><td><b>Fodder Value/Kg: </b></td><td>" + QString::number(mCropFodderValue) +
   "</td></tr>";
  //myString+="<tr><td><b>FodderEnergyType: </b></td><td>" + myCropFodderEnergyType + "</td></tr>";
  //myString+="<tr><td><b>AreaUnits: </b></td><td>" + myUnits + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

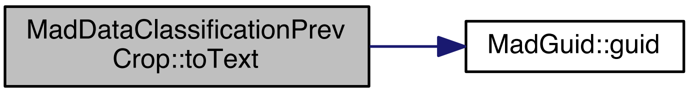
#### 6.6.3.20 QString MadDataClassificationPrevCrop::toText ( )

Return a plain text representation of this layer

Definition at line 189 of file maddataclassificationprevcrop.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  //myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.6.3.21 QString MadDataClassificationPrevCrop::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

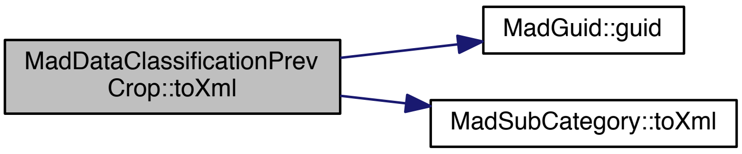
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

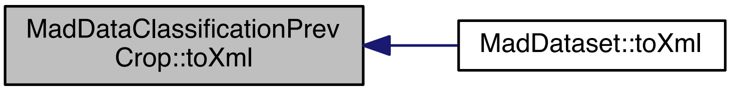
Definition at line 151 of file maddataclassificationprevcrop.cpp.

```
{
  QString myString;
  myString+=QString("  <prevcrop guid=\"" + guid() + "\">\n");

  myString+=QString("    <crop>\n");
  myString+=mCrop.toXml();
  myString+=QString("    </crop>\n");

  myString+=QString("    <sowingdate>\n");
  myString+=mSowingDate.toXml();
  myString+=QString("    </sowingdate>\n");

  myString+=QString("    <harvestdate>\n");
  myString+=mHarvestDate.toXml();
  myString+=QString("    </harvestdate>\n");

  myString+=QString("    <yield>\n");
  myString+=mYield.toXml();
  myString+=QString("    </yield>\n");

  myString+=QString("    <residuemgmt>\n");
  myString+=mResidueMgmt.toXml();
  myString+=QString("    </residuemgmt>\n");

  myString+=QString("    <fertilisation>\n");
  myString+=mFertilisation.toXml();
  myString+=QString("    </fertilisation>\n");

  myString+=QString("    <irrigation>\n");
  myString+=mIrrigation.toXml();
  myString+=QString("    </irrigation>\n");

  myString+=QString("  </prevcrop>\n");
  return myString;

}
```

Here is the call graph for this function:



Here is the caller graph for this function:



### 6.6.3.22 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

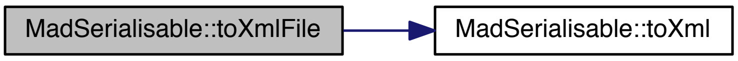
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:

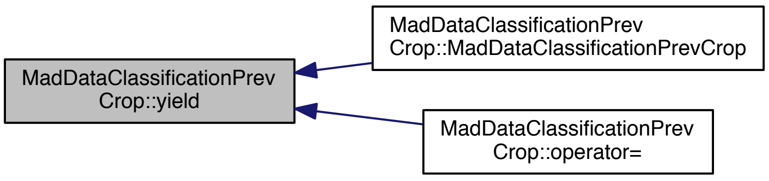


#### 6.6.3.23 MadSubCategory MadDataClassificationPrevCrop::yield ( ) const

Definition at line 79 of file maddataclassificationprevcrop.cpp.

```
{
  return mYield;
}
```

Here is the caller graph for this function:


MadDataClassificationPrev Crop::yield
MadDataClassificationPrev Crop::MadDataClassificationPrevCrop
MadDataClassificationPrev Crop::operator=

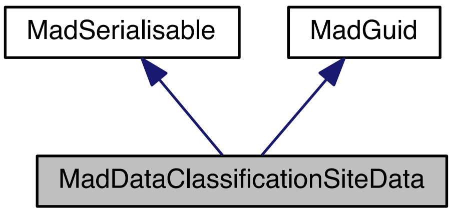
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationprevcrop.-h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationprevcrop.-cpp

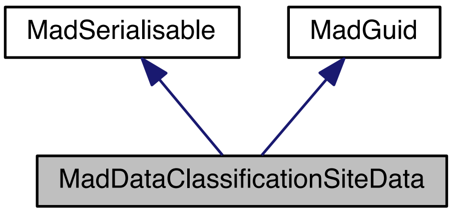
## 6.7 MadDataClassificationSiteData Class Reference

```
#include <maddataclassificationsitedata.h>
```

Inheritance diagram for MadDataClassificationSiteData:



Collaboration diagram for MadDataClassificationSiteData:



### Public Member Functions

- MadDataClassificationSiteData ()
- MadDataClassificationSiteData (const MadDataClassificationSiteData &theData)
- MadDataClassificationSiteData & operator= (const MadDataClassificationSiteData &theData)
- MadSubCategory latitude () const
- MadSubCategory longitude () const
- MadSubCategory altitude () const
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- void setLatitude (MadSubCategory theData)
- void setLongitude (MadSubCategory theData)
- void setAltitude (MadSubCategory theData)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*

- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.7.1 Detailed Description

Definition at line 36 of file maddataclassificationsitedata.h.

### 6.7.2 Constructor & Destructor Documentation

#### 6.7.2.1 MadDataClassificationSiteData::MadDataClassificationSiteData ( )

Definition at line 33 of file maddataclassificationsitedata.cpp.

```
                                                              : MadSerialisable(),
    MadGuid()
{
  setGuid();
}
```

Here is the call graph for this function:

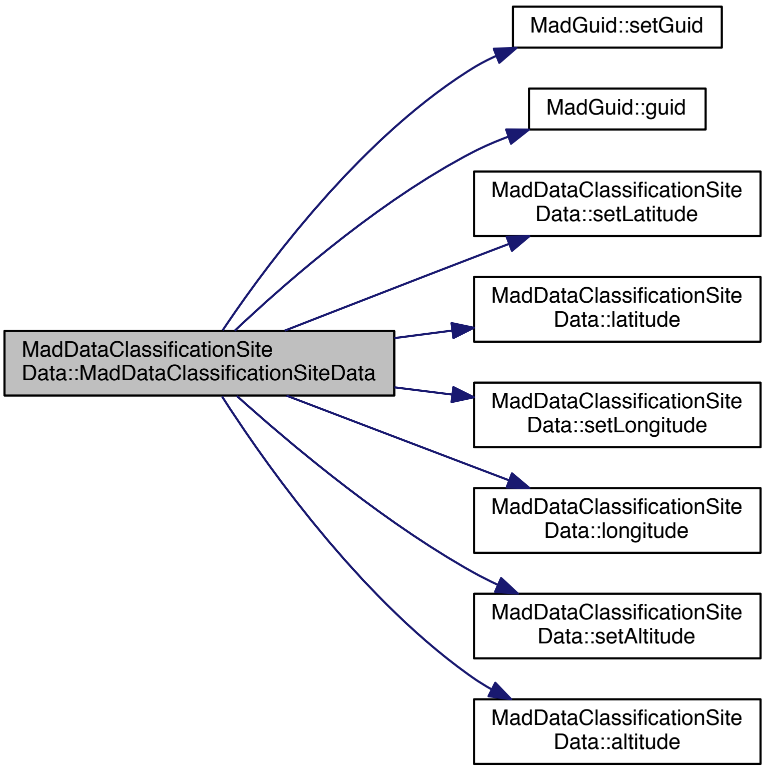
MadDataClassificationSite Data::MadDataClassificationSiteData
MadGuid::setGuid

#### 6.7.2.2 MadDataClassificationSiteData::MadDataClassificationSiteData ( const MadDataClassificationSiteData & *theData* )

Definition at line 38 of file maddataclassificationsitedata.cpp.

```
{
  setGuid(theData.guid());
  setLatitude(theData.latitude());
  setLongitude(theData.longitude());
  setAltitude(theData.altitude());
}
```

Here is the call graph for this function:


MadGuid::setGuid
MadGuid::guid
MadDataClassificationSiteData::setLatitude
MadDataClassificationSiteData::latitude
MadDataClassificationSiteData::MadDataClassificationSiteData
MadDataClassificationSiteData::setLongitude
MadDataClassificationSiteData::longitude
MadDataClassificationSiteData::setAltitude
MadDataClassificationSiteData::altitude

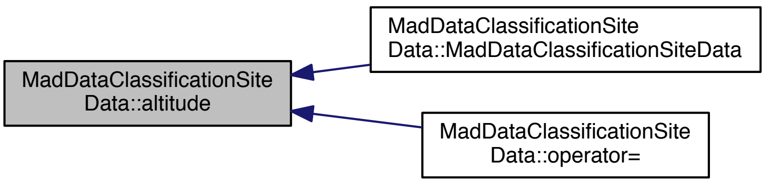
### 6.7.3 Member Function Documentation

#### 6.7.3.1 MadSubCategory MadDataClassificationSiteData::altitude ( ) const

Definition at line 65 of file maddataclassificationsitedata.cpp.

```
{
  return mAltitude;
}
```

Here is the caller graph for this function:


MadDataClassificationSiteData::altitude
MadDataClassificationSiteData::MadDataClassificationSiteData
MadDataClassificationSiteData::operator=

#### 6.7.3.2 bool MadDataClassificationSiteData::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

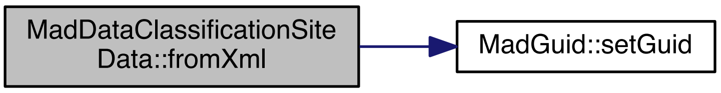
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 86 of file maddataclassificationsitedata.cpp.

```
{
    QDomDocument myDocument("mydocument");
    myDocument.setContent(theXml);
    QDomElement myTopElement = myDocument.firstChildElement("site");
    if (myTopElement.isNull())
    {
        //TODO - just make this a warning
        qDebug("the top element couldn't be found!");
        setGuid(myTopElement.attribute("guid"));
        //mName=MadUtils::xmlDecode(myTopElement.firstChildElement("name").text());
        //mDescription=MadUtils::xmlDecode(myTopElement.firstChildElement("description").text());
        //mImageFile=QString(myTopElement.firstChildElement("imageFile").text());
        return true;
     }
     else
     return false;
}
```

Here is the call graph for this function:



#### 6.7.3.3 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

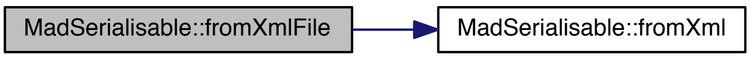
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



### 6.7.3.4 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

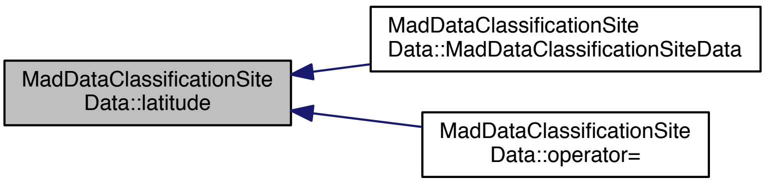
```
{
    return mGuid;
}
```

### 6.7.3.5 MadSubCategory MadDataClassificationSiteData::latitude ( ) const

Definition at line 57 of file maddataclassificationsitedata.cpp.

```
{
  return mLatitude;
}
```

Here is the caller graph for this function:

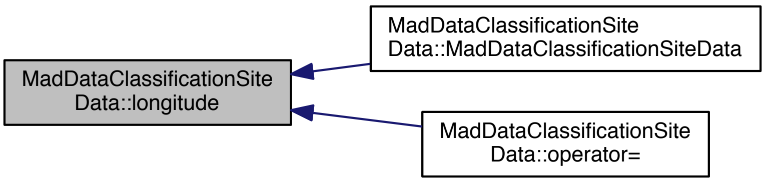

MadDataClassificationSite Data::latitude
MadDataClassificationSite Data::MadDataClassificationSiteData
MadDataClassificationSite Data::operator=

### 6.7.3.6 MadSubCategory MadDataClassificationSiteData::longitude ( ) const

Definition at line 61 of file maddataclassificationsitedata.cpp.

```
{
  return mLongitude;
}
```

Here is the caller graph for this function:

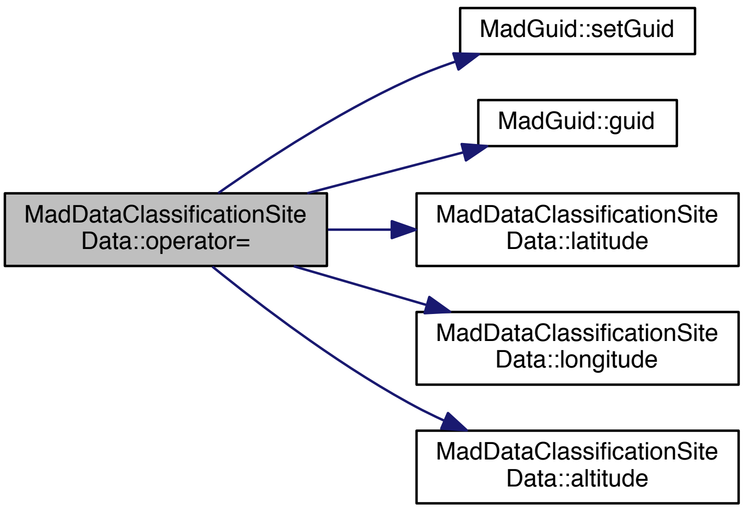

MadDataClassificationSite Data::longitude
MadDataClassificationSite Data::MadDataClassificationSiteData
MadDataClassificationSite Data::operator=

### 6.7.3.7 MadDataClassificationSiteData & MadDataClassificationSiteData::operator= ( const MadDataClassificationSiteData & *theData* )

Definition at line 46 of file maddataclassificationsitedata.cpp.

```
{
  // gracefully handles self assignment
  if (this == &theData) return *this;
  setGuid(theData.guid());
  mLatitude=theData.latitude();
  mLongitude=theData.longitude();
  mAltitude=theData.altitude();
  return *this;
}
```

Here is the call graph for this function:

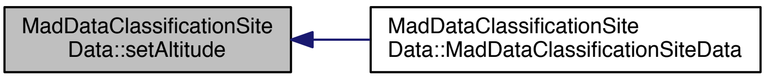

MadGuid::setGuid
MadGuid::guid
MadDataClassificationSite Data::operator=
MadDataClassificationSite Data::latitude
MadDataClassificationSite Data::longitude
MadDataClassificationSite Data::altitude

### 6.7.3.8 void MadDataClassificationSiteData::setAltitude ( MadSubCategory *theData* )

Definition at line 81 of file maddataclassificationsitedata.cpp.

```
{
  mAltitude = theData;
}
```

Here is the caller graph for this function:


MadDataClassificationSite Data::setAltitude
MadDataClassificationSite Data::MadDataClassificationSiteData

### 6.7.3.9 void MadGuid::setGuid ( QString *theGuid* = "" ) [inherited]

MadGuid::setGuid.

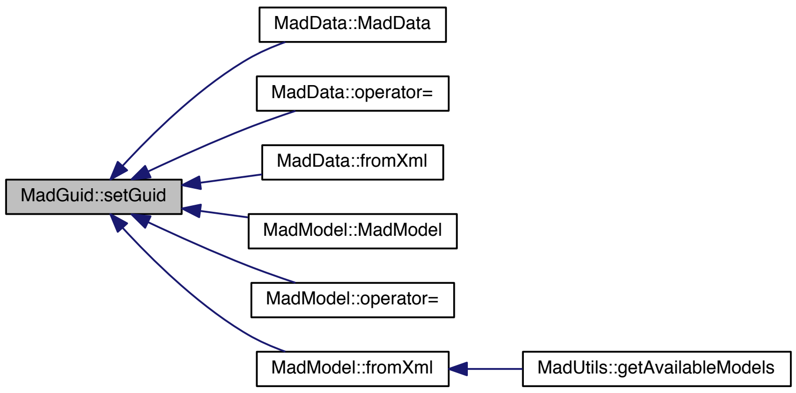
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:

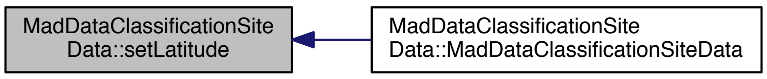


#### 6.7.3.10 void MadDataClassificationSiteData::setLatitude ( MadSubCategory *theData* )

Definition at line 71 of file maddataclassificationsitedata.cpp.

```
{
  mLatitude = theData;
}
```

Here is the caller graph for this function:

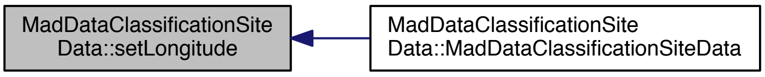

MadDataClassificationSite Data::setLatitude
MadDataClassificationSite Data::MadDataClassificationSiteData

#### 6.7.3.11 void MadDataClassificationSiteData::setLongitude ( MadSubCategory *theData* )

Definition at line 76 of file maddataclassificationsitedata.cpp.

```
{
  mLongitude = theData;
}
```

Here is the caller graph for this function:


MadDataClassificationSite Data::setLongitude
MadDataClassificationSite Data::MadDataClassificationSiteData

#### 6.7.3.12 QString MadDataClassificationSiteData::toHtml ( )

Return a html text representation of this layer

Definition at line 136 of file maddataclassificationsitedata.cpp.

```
{
  QString myString;
  //myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  //myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";

  //
  // the following shows example of how to do a couple of things
  //

  //myString+="<tr><td><b>Cals/Kg: </b></td><td>" + QString::number(mCropCalories) + "</td></tr>";
  //QString myCropFodderEnergyType = (mCropFodderEnergyType==0) ? "KCalories" : "TDN";
  //QString myUnits = (mAreaUnits==0) ? "Dunum" : "Hectare";
  //myString+="<tr><td><b>Fodder (kg/" + myUnits + "): </b></td><td>" +
   QString::number(mCropFodderProduction) + "</td></tr>";
  //myString+="<tr><td><b>Fodder Value/Kg: </b></td><td>" + QString::number(mCropFodderValue) +
   "</td></tr>";
  //myString+="<tr><td><b>FodderEnergyType: </b></td><td>" + myCropFodderEnergyType + "</td></tr>";
  //myString+="<tr><td><b>AreaUnits: </b></td><td>" + myUnits + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

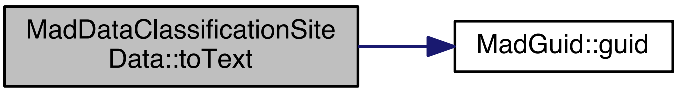
#### 6.7.3.13 QString MadDataClassificationSiteData::toText ( )

Return a plain text representation of this layer

Definition at line 127 of file maddataclassificationsitedata.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  //myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.7.3.14 QString MadDataClassificationSiteData::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

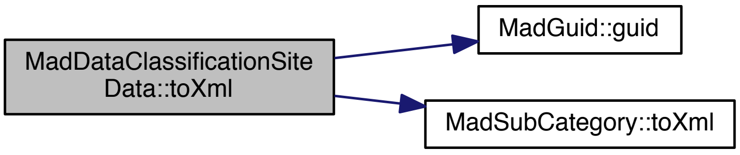
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

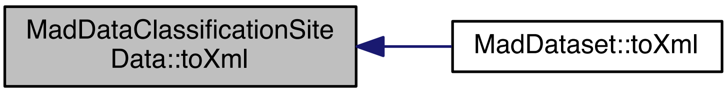
Definition at line 105 of file maddataclassificationsitedata.cpp.

```
{
  QString myString;
  myString+=QString("  <site guid=\"" + guid() + "\">\n");

  myString+=QString("     <latitude>\n");
  myString+=mLatitude.toXml();
  myString+=QString("     </latitude>\n");

  myString+=QString("     <longitude>\n");
  myString+=mLongitude.toXml();
  myString+=QString("     </longitude>\n");

  myString+=QString("     <altitude>\n");
  myString+=mAltitude.toXml();
  myString+=QString("     </altitude>\n");

  myString+=QString("  </site>\n");
  return myString;

}
```

Here is the call graph for this function:



Here is the caller graph for this function:



#### 6.7.3.15 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

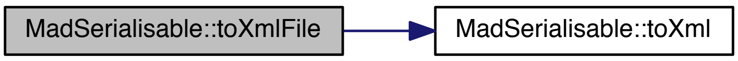
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



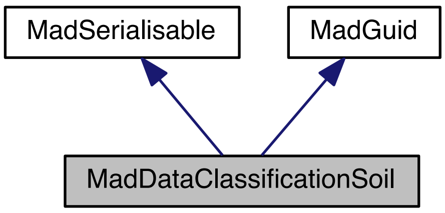
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationsitedata.-h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationsitedata.-cpp

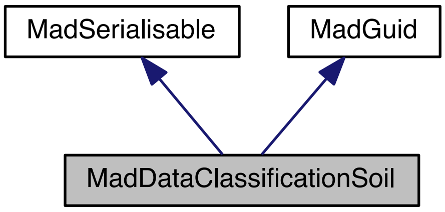
## 6.8 MadDataClassificationSoil Class Reference

```
#include <maddataclassificationsoil.h>
```

Inheritance diagram for MadDataClassificationSoil:



Collaboration diagram for MadDataClassificationSoil:



### Public Member Functions

- MadDataClassificationSoil ()
- MadDataClassificationSoil (const MadDataClassificationSoil &theData)
- MadDataClassificationSoil & operator= (const MadDataClassificationSoil &theData)
- MadSubCategory carbonOrganic () const
- MadSubCategory nitrogenOrganic () const
- MadSubCategory texture () const
- MadSubCategory bulkDensity () const
- MadSubCategory fieldCapacityMeas () const

- MadSubCategory wiltingPointMeas () const
- MadSubCategory pfCurve () const
- MadSubCategory hydrCondCurve () const
- MadSubCategory pH () const
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- void setCarbonOrganic (MadSubCategory theData)
- void setNitrogenOrganic (MadSubCategory theData)
- void setTexture (MadSubCategory theData)
- void setBulkDensity (MadSubCategory theData)
- void setFieldCapacityMeas (MadSubCategory theData)
- void setWiltingPointMeas (MadSubCategory theData)
- void setPfCurve (MadSubCategory theData)
- void setHydrCondCurve (MadSubCategory theData)
- void setPh (MadSubCategory theData)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.8.1 Detailed Description

Definition at line 34 of file maddataclassificationsoil.h.

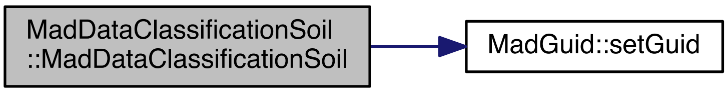
### 6.8.2 Constructor & Destructor Documentation

#### 6.8.2.1 MadDataClassificationSoil::MadDataClassificationSoil ( )

Definition at line 33 of file maddataclassificationsoil.cpp.

```
                                                         : MadSerialisable(),
      MadGuid()
{
  setGuid();
}
```

Here is the call graph for this function:

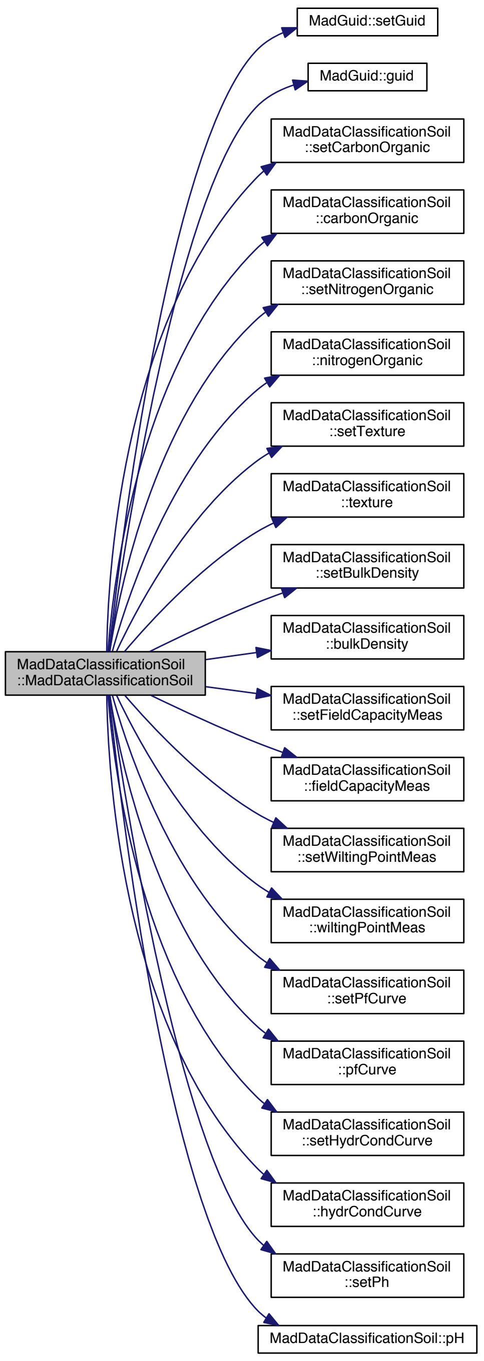


#### 6.8.2.2 MadDataClassificationSoil::MadDataClassificationSoil ( const MadDataClassificationSoil & *theData* )

Definition at line 38 of file maddataclassificationsoil.cpp.

```
{
  setGuid(theData.guid());
  setCarbonOrganic(theData.carbonOrganic());
  setNitrogenOrganic(theData.nitrogenOrganic());
  setTexture(theData.texture());
  setBulkDensity(theData.bulkDensity());
  setFieldCapacityMeas(theData.fieldCapacityMeas());
  setWiltingPointMeas(theData.wiltingPointMeas());
  setPfCurve(theData.pfCurve());
  setHydrCondCurve(theData.hydrCondCurve());
  setPh(theData.pH());
}
```

Here is the call graph for this function:


MadDataClassificationSoil ::MadDataClassificationSoil
MadGuid::setGuid
MadGuid::guid
MadDataClassificationSoil ::setCarbonOrganic
MadDataClassificationSoil ::carbonOrganic
MadDataClassificationSoil ::setNitrogenOrganic
MadDataClassificationSoil ::nitrogenOrganic
MadDataClassificationSoil ::setTexture
MadDataClassificationSoil ::texture
MadDataClassificationSoil ::setBulkDensity
MadDataClassificationSoil ::bulkDensity
MadDataClassificationSoil ::setFieldCapacityMeas
MadDataClassificationSoil ::fieldCapacityMeas
MadDataClassificationSoil ::setWiltingPointMeas
MadDataClassificationSoil ::wiltingPointMeas
MadDataClassificationSoil ::setPfCurve
MadDataClassificationSoil ::pfCurve
MadDataClassificationSoil ::setHydrCondCurve
MadDataClassificationSoil ::hydrCondCurve
MadDataClassificationSoil ::setPh
MadDataClassificationSoil::pH

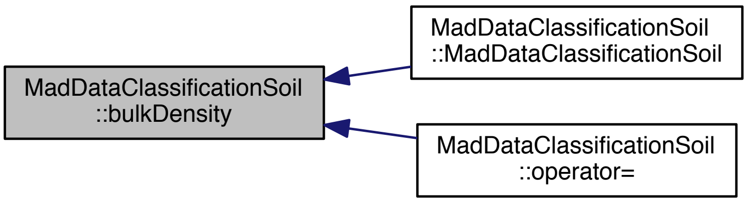
### 6.8.3 Member Function Documentation

#### 6.8.3.1 MadSubCategory MadDataClassificationSoil::bulkDensity ( ) const

Definition at line 82 of file maddataclassificationsoil.cpp.

```
{
  return mBulkDensity;
}
```

Here is the caller graph for this function:

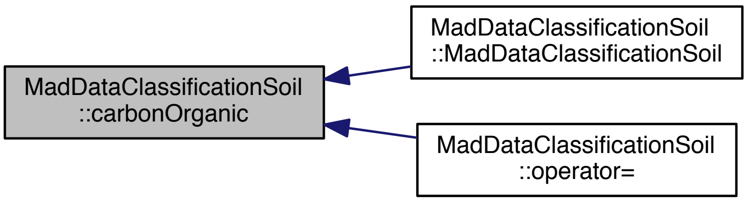

MadDataClassificationSoil ::bulkDensity
MadDataClassificationSoil ::MadDataClassificationSoil
MadDataClassificationSoil ::operator=

#### 6.8.3.2 MadSubCategory MadDataClassificationSoil::carbonOrganic ( ) const

Definition at line 70 of file maddataclassificationsoil.cpp.

```
{
  return mCarbonOrganic;
}
```

Here is the caller graph for this function:

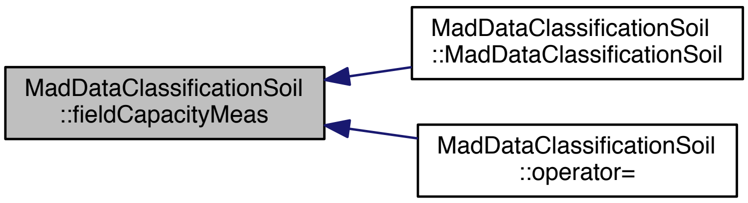

MadDataClassificationSoil ::carbonOrganic
MadDataClassificationSoil ::MadDataClassificationSoil
MadDataClassificationSoil ::operator=

#### 6.8.3.3 MadSubCategory MadDataClassificationSoil::fieldCapacityMeas ( ) const

Definition at line 86 of file maddataclassificationsoil.cpp.

```
{
  return mFieldCapacityMeas;
}
```

Here is the caller graph for this function:


MadDataClassificationSoil ::fieldCapacityMeas
MadDataClassificationSoil ::MadDataClassificationSoil
MadDataClassificationSoil ::operator=

#### 6.8.3.4 bool MadDataClassificationSoil::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

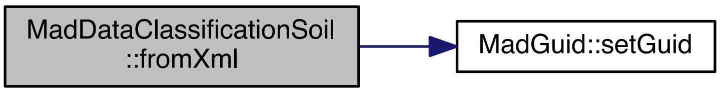
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 153 of file maddataclassificationsoil.cpp.

```
{
    QDomDocument myDocument("mydocument");
    myDocument.setContent(theXml);
    QDomElement myTopElement = myDocument.firstChildElement("soil");
    if (myTopElement.isNull())
    {
        //TODO - just make this a warning
        qDebug("the top element couldn't be found!");
        setGuid(myTopElement.attribute("guid"));
        //mName=MadUtils::xmlDecode(myTopElement.firstChildElement("name").text());
        //mDescription=MadUtils::xmlDecode(myTopElement.firstChildElement("description").text());
        //mImageFile=QString(myTopElement.firstChildElement("imageFile").text());
        return true;
    }
    else
    return false;
}
```

Here is the call graph for this function:



#### 6.8.3.5 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

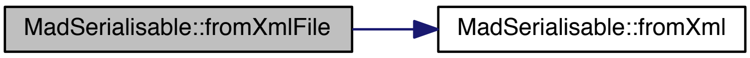
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



#### 6.8.3.6 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

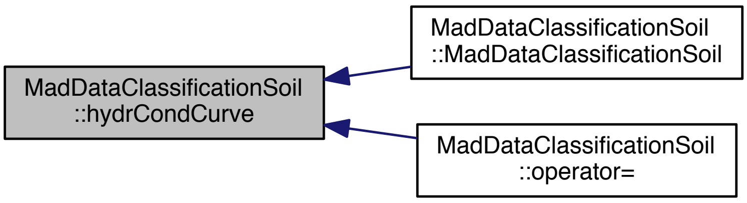
```
{
    return mGuid;
}
```

#### 6.8.3.7 MadSubCategory MadDataClassificationSoil::hydrCondCurve ( ) const

Definition at line 98 of file maddataclassificationsoil.cpp.

```
{
  return mHydrCondCurve;
}
```

Here is the caller graph for this function:

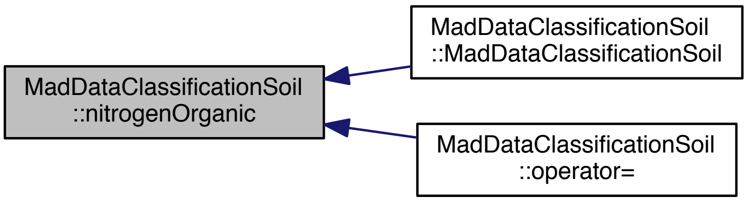

MadDataClassificationSoil ::hydrCondCurve
MadDataClassificationSoil ::MadDataClassificationSoil
MadDataClassificationSoil ::operator=

#### 6.8.3.8 MadSubCategory MadDataClassificationSoil::nitrogenOrganic ( ) const

Definition at line 74 of file maddataclassificationsoil.cpp.

```
{
  return mNitrogenOrganic;
}
```

Here is the caller graph for this function:


MadDataClassificationSoil ::nitrogenOrganic
MadDataClassificationSoil ::MadDataClassificationSoil
MadDataClassificationSoil ::operator=

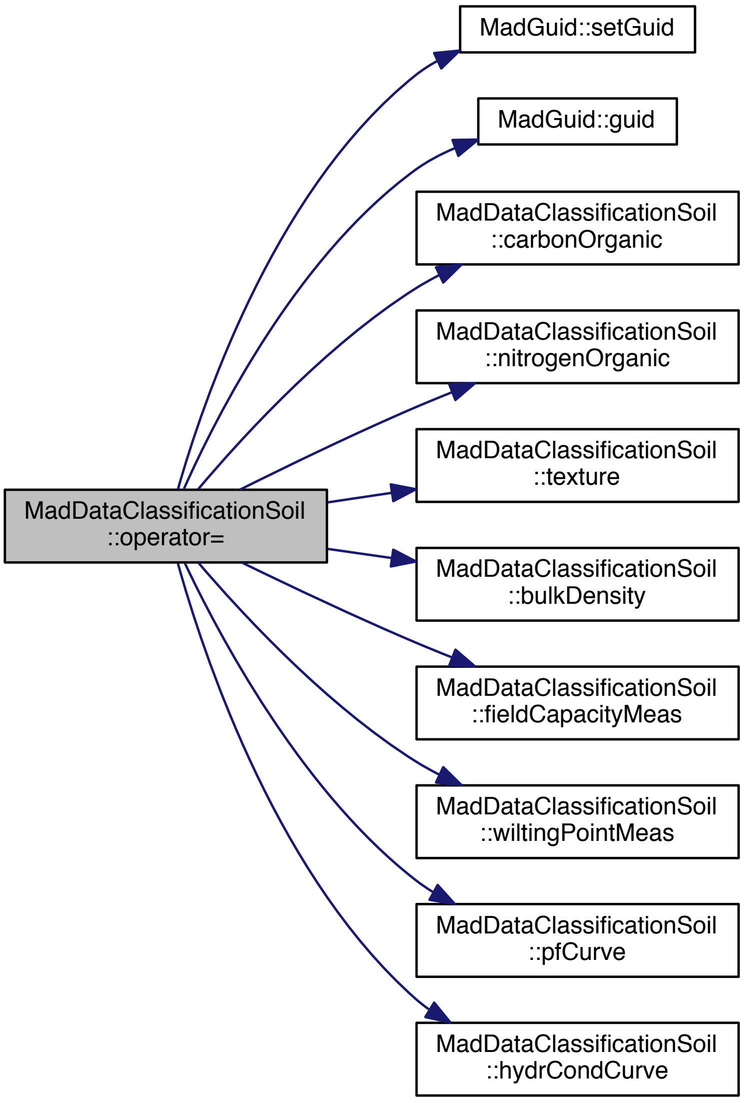
#### 6.8.3.9 MadDataClassificationSoil & MadDataClassificationSoil::operator= ( const MadDataClassificationSoil & *theData* )

Definition at line 52 of file maddataclassificationsoil.cpp.

```
{
  // gracefully handles self assignment
  if (this == &theData) return *this;
  setGuid(theData.guid());
```

```
    mCarbonOrganic=theData.carbonOrganic();
    mNitrogenOrganic=theData.nitrogenOrganic();
    mTexture=theData.texture();
    mBulkDensity=theData.bulkDensity();
    mFieldCapacityMeas=theData.fieldCapacityMeas();
    mWiltingPointMeas=theData.wiltingPointMeas();
    mPfCurve=theData.pfCurve();
    mHydrCondCurve=theData.hydrCondCurve();
    mPH=theData.mPH;
    return *this;
}
```

Here is the call graph for this function:

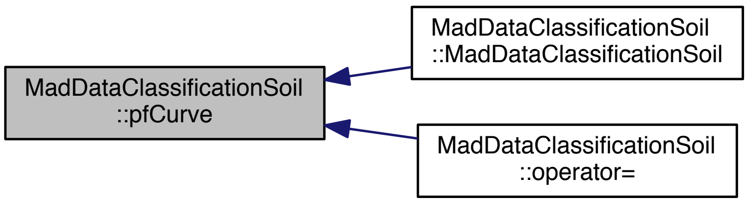

MadGuid::setGuid
MadGuid::guid
MadDataClassificationSoil ::carbonOrganic
MadDataClassificationSoil ::nitrogenOrganic
MadDataClassificationSoil ::texture
MadDataClassificationSoil ::operator=
MadDataClassificationSoil ::bulkDensity
MadDataClassificationSoil ::fieldCapacityMeas
MadDataClassificationSoil ::wiltingPointMeas
MadDataClassificationSoil ::pfCurve
MadDataClassificationSoil ::hydrCondCurve

#### 6.8.3.10 MadSubCategory MadDataClassificationSoil::pfCurve ( ) const

Definition at line 94 of file maddataclassificationsoil.cpp.

```
{
  return mPfCurve;
}
```

Here is the caller graph for this function:

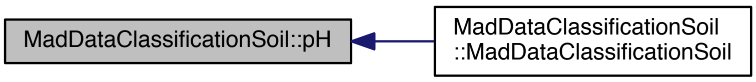

MadDataClassificationSoil
::pfCurve
MadDataClassificationSoil
::MadDataClassificationSoil
MadDataClassificationSoil
::operator=

#### 6.8.3.11 MadSubCategory MadDataClassificationSoil::pH ( ) const

Definition at line 102 of file maddataclassificationsoil.cpp.

```
{
  return mPH;
}
```

Here is the caller graph for this function:

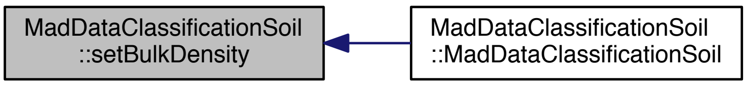

MadDataClassificationSoil::pH
MadDataClassificationSoil
::MadDataClassificationSoil

#### 6.8.3.12 void MadDataClassificationSoil::setBulkDensity ( MadSubCategory *theData* )

Definition at line 122 of file maddataclassificationsoil.cpp.

```
{
  mBulkDensity = theData;
}
```

Here is the caller graph for this function:

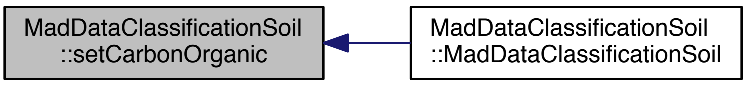

MadDataClassificationSoil ::setBulkDensity
MadDataClassificationSoil ::MadDataClassificationSoil

#### 6.8.3.13 void MadDataClassificationSoil::setCarbonOrganic ( MadSubCategory *theData* )

Definition at line 107 of file maddataclassificationsoil.cpp.

```
{
  mCarbonOrganic = theData;
}
```

Here is the caller graph for this function:

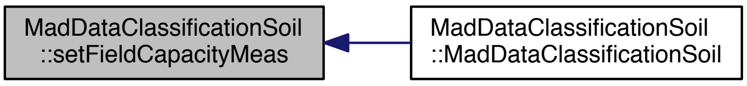

MadDataClassificationSoil ::setCarbonOrganic
MadDataClassificationSoil ::MadDataClassificationSoil

#### 6.8.3.14 void MadDataClassificationSoil::setFieldCapacityMeas ( MadSubCategory *theData* )

Definition at line 127 of file maddataclassificationsoil.cpp.

```
{
  mFieldCapacityMeas = theData;
}
```

Here is the caller graph for this function:


MadDataClassificationSoil ::setFieldCapacityMeas
MadDataClassificationSoil ::MadDataClassificationSoil

#### 6.8.3.15 void MadGuid::setGuid ( QString *theGuid* = " " ) [inherited]

MadGuid::setGuid.

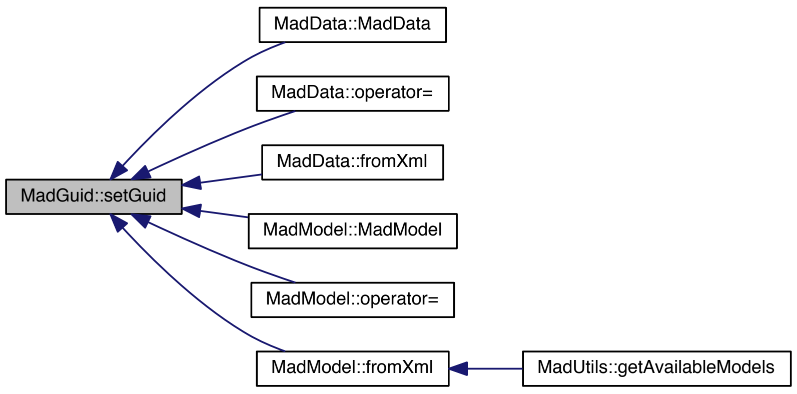
**Parameters**

|  |  |
|---|---|
| *theGuid* |  |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:

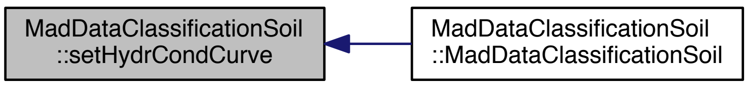


#### 6.8.3.16 void MadDataClassificationSoil::setHydrCondCurve ( MadSubCategory *theData* )

Definition at line 142 of file maddataclassificationsoil.cpp.

```
{
  mHydrCondCurve = theData;
}
```

Here is the caller graph for this function:

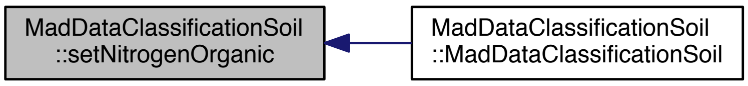

MadDataClassificationSoil ::setHydrCondCurve
MadDataClassificationSoil ::MadDataClassificationSoil

#### 6.8.3.17 void MadDataClassificationSoil::setNitrogenOrganic ( MadSubCategory *theData* )

Definition at line 112 of file maddataclassificationsoil.cpp.

```
{
  mNitrogenOrganic = theData;
}
```

Here is the caller graph for this function:

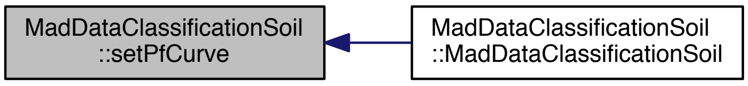

MadDataClassificationSoil ::setNitrogenOrganic
MadDataClassificationSoil ::MadDataClassificationSoil

#### 6.8.3.18 void MadDataClassificationSoil::setPfCurve ( MadSubCategory *theData* )

Definition at line 137 of file maddataclassificationsoil.cpp.

```
{
  mPfCurve = theData;
}
```

Here is the caller graph for this function:

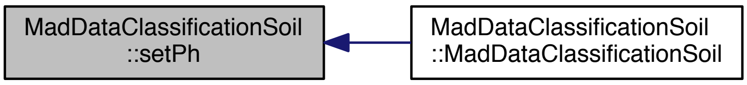

MadDataClassificationSoil ::setPfCurve
MadDataClassificationSoil ::MadDataClassificationSoil

#### 6.8.3.19 void MadDataClassificationSoil::setPh ( MadSubCategory *theData* )

Definition at line 147 of file maddataclassificationsoil.cpp.

```
{
  mPH = theData;
}
```

Here is the caller graph for this function:

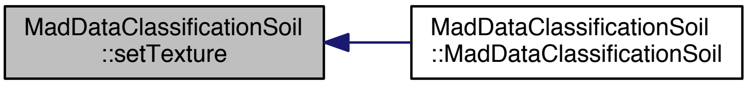

MadDataClassificationSoil ::setPh
MadDataClassificationSoil ::MadDataClassificationSoil

#### 6.8.3.20 void MadDataClassificationSoil::setTexture ( **MadSubCategory** *theData* )

Definition at line 117 of file maddataclassificationsoil.cpp.

```
{
  mTexture = theData;
}
```

Here is the caller graph for this function:

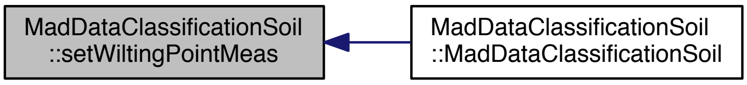

MadDataClassificationSoil ::setTexture
MadDataClassificationSoil ::MadDataClassificationSoil

#### 6.8.3.21 void MadDataClassificationSoil::setWiltingPointMeas ( **MadSubCategory** *theData* )

Definition at line 132 of file maddataclassificationsoil.cpp.

```
{
  mWiltingPointMeas = theData;
}
```

Here is the caller graph for this function:

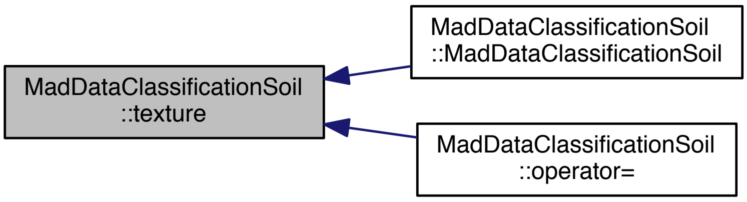

MadDataClassificationSoil ::setWiltingPointMeas
MadDataClassificationSoil ::MadDataClassificationSoil

#### 6.8.3.22 MadSubCategory MadDataClassificationSoil::texture ( ) const

Definition at line 78 of file maddataclassificationsoil.cpp.

```
{
  return mTexture;
}
```

Here is the caller graph for this function:


MadDataClassificationSoil ::texture
MadDataClassificationSoil ::MadDataClassificationSoil
MadDataClassificationSoil ::operator=

#### 6.8.3.23 QString MadDataClassificationSoil::toHtml ( )

Return a html text representation of this layer

Definition at line 227 of file maddataclassificationsoil.cpp.

```
{
  QString myString;
  //myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  //myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";

  //
  // the following shows example of how to do a couple of things
  //

  //myString+="<tr><td><b>Cals/Kg: </b></td><td>" + QString::number(mCropCalories) + "</td></tr>";
  //QString myCropFodderEnergyType = (mCropFodderEnergyType==0) ? "KCalories" : "TDN";
  //QString myUnits = (mAreaUnits==0) ? "Dunum" : "Hectare";
  //myString+="<tr><td><b>Fodder (kg/" + myUnits + "): </b></td><td>" +
   QString::number(mCropFodderProduction) + "</td></tr>";
  //myString+="<tr><td><b>Fodder Value/Kg: </b></td><td>" + QString::number(mCropFodderValue) +
   "</td></tr>";
  //myString+="<tr><td><b>FodderEnergyType: </b></td><td>" + myCropFodderEnergyType + "</td></tr>";
  //myString+="<tr><td><b>AreaUnits: </b></td><td>" + myUnits + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

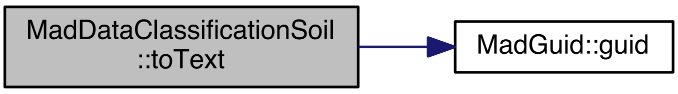
#### 6.8.3.24 QString MadDataClassificationSoil::toText ( )

Return a plain text representation of this layer

Definition at line 218 of file maddataclassificationsoil.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  //myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.8.3.25 QString MadDataClassificationSoil::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

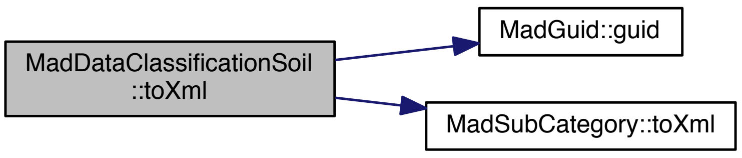
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

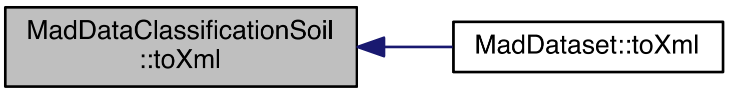
Definition at line 172 of file maddataclassificationsoil.cpp.

```
{
  QString myString;
  myString+=QString("  <soil guid=\"" + guid() + "\">\n");

  myString+=QString("    <corg>\n");
  myString+=mCarbonOrganic.toXml();
  myString+=QString("    </corg>\n");

  myString+=QString("    <norg>\n");
  myString+=mNitrogenOrganic.toXml();
  myString+=QString("    </norg>\n");

  myString+=QString("    <texture>\n");
  myString+=mTexture.toXml();
  myString+=QString("    </texture>\n");

  myString+=QString("    <bulkdensity>\n");
  myString+=mBulkDensity.toXml();
  myString+=QString("    </bulkdensity>\n");

  myString+=QString("    <fieldcapacity>\n");
  myString+=mFieldCapacityMeas.toXml();
  myString+=QString("    </fieldcapacity>\n");

  myString+=QString("    <wiltingpoint>\n");
  myString+=mWiltingPointMeas.toXml();
  myString+=QString("    </wiltingpoint>\n");

  myString+=QString("    <pfcurve>\n");
  myString+=mPfCurve.toXml();
  myString+=QString("    </pfcurve>\n");

  myString+=QString("    <hydrcondcurve>\n");
  myString+=mHydrCondCurve.toXml();
  myString+=QString("    </hydrcondcurve>\n");

  myString+=QString("    <ph>\n");
  myString+=mPH.toXml();
  myString+=QString("    </ph>\n");

  myString+=QString("  </soil>\n");
  return myString;

}
```

Here is the call graph for this function:



Here is the caller graph for this function:



#### 6.8.3.26 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

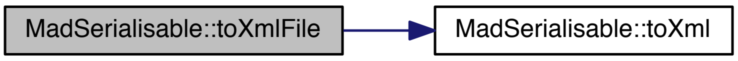
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:

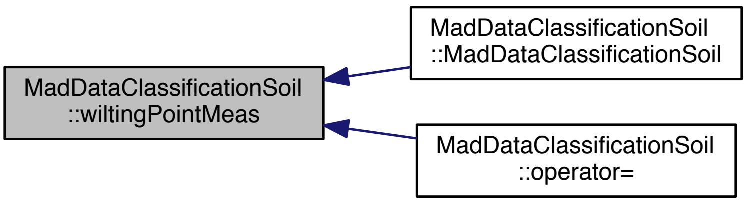


### 6.8.3.27 MadSubCategory MadDataClassificationSoil::wiltingPointMeas ( ) const

Definition at line 90 of file maddataclassificationsoil.cpp.

```
{
  return mWiltingPointMeas;
}
```

Here is the caller graph for this function:


MadDataClassificationSoil ::wiltingPointMeas
MadDataClassificationSoil ::MadDataClassificationSoil
MadDataClassificationSoil ::operator=

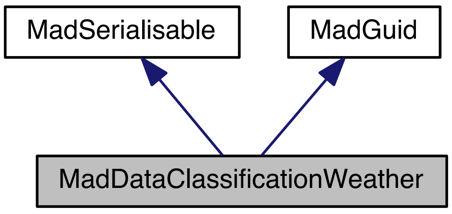
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationsoil.-h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationsoil.-cpp

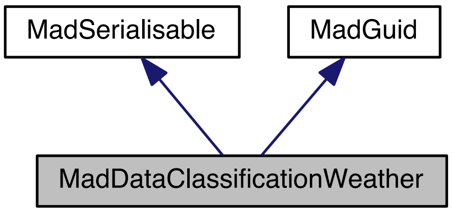
## 6.9 MadDataClassificationWeather Class Reference

```
#include <maddataclassificationweather.h>
```

Inheritance diagram for MadDataClassificationWeather:



Collaboration diagram for MadDataClassificationWeather:



## Public Member Functions

- MadDataClassificationWeather ()
- MadDataClassificationWeather (const MadDataClassificationWeather &theData)
- MadDataClassificationWeather & operator= (const MadDataClassificationWeather &theData)
- bool minData () const
- MadSubCategory precipitation () const
- MadSubCategory tAve () const
- MadSubCategory tMin () const
- MadSubCategory tMax () const
- MadSubCategory relativeHumidity () const
- MadSubCategory windSpeed () const
- MadSubCategory globalRadiation () const
- MadSubCategory sunshineHours () const
- MadSubCategory leafWetness () const
- MadSubCategory soilTemp () const
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- void setMinData (bool theBool)
- void setPrecipitation (MadSubCategory theData)

- void setTAve (MadSubCategory theData)
- void setTMin (MadSubCategory theData)
- void setTMax (MadSubCategory theData)
- void setRelativeHumidity (MadSubCategory theData)
- void setWindSpeed (MadSubCategory theData)
- void setGlobalRadiation (MadSubCategory theData)
- void setSunshineHours (MadSubCategory theData)
- void setLeafWetness (MadSubCategory theData)
- void setSoilTemp (MadSubCategory theData)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.9.1 Detailed Description

Definition at line 35 of file maddataclassificationweather.h.

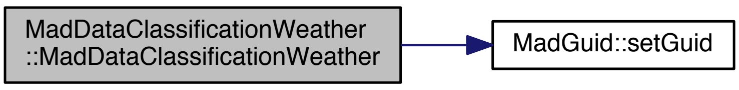
### 6.9.2 Constructor & Destructor Documentation

#### 6.9.2.1 MadDataClassificationWeather::MadDataClassificationWeather ( )

Definition at line 33 of file maddataclassificationweather.cpp.

```
33                                                              : MadSerialisable(),
      MadGuid()
34 {
35    setGuid();
36 }
```

Here is the call graph for this function:

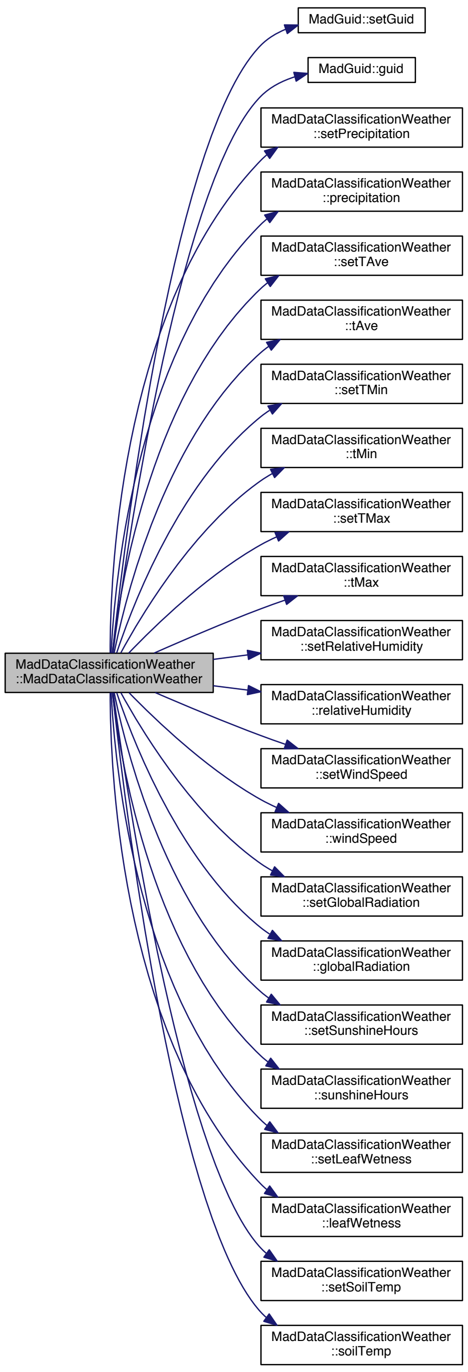


#### 6.9.2.2 MadDataClassificationWeather::MadDataClassificationWeather ( const MadDataClassificationWeather & *theData* )

Definition at line 38 of file maddataclassificationweather.cpp.

```
{
  setGuid(theData.guid());
  setPrecipitation(theData.precipitation());
  setTAve(theData.tAve());
  setTMin(theData.tMin());
  setTMax(theData.tMax());
  setRelativeHumidity(theData.relativeHumidity());
  setWindSpeed(theData.windSpeed());
  setGlobalRadiation(theData.globalRadiation());
  setSunshineHours(theData.sunshineHours());
  setLeafWetness(theData.leafWetness());
  setSoilTemp(theData.soilTemp());
}
```

Here is the call graph for this function:


MadDataClassificationWeather ::MadDataClassificationWeather
MadGuid::setGuid
MadGuid::guid
MadDataClassificationWeather ::setPrecipitation
MadDataClassificationWeather ::precipitation
MadDataClassificationWeather ::setTAve
MadDataClassificationWeather ::tAve
MadDataClassificationWeather ::setTMin
MadDataClassificationWeather ::tMin
MadDataClassificationWeather ::setTMax
MadDataClassificationWeather ::tMax
MadDataClassificationWeather ::setRelativeHumidity
MadDataClassificationWeather ::relativeHumidity
MadDataClassificationWeather ::setWindSpeed
MadDataClassificationWeather ::windSpeed
MadDataClassificationWeather ::setGlobalRadiation
MadDataClassificationWeather ::globalRadiation
MadDataClassificationWeather ::setSunshineHours
MadDataClassificationWeather ::sunshineHours
MadDataClassificationWeather ::setLeafWetness
MadDataClassificationWeather ::leafWetness
MadDataClassificationWeather ::setSoilTemp
MadDataClassificationWeather ::soilTemp

### 6.9.3 Member Function Documentation

#### 6.9.3.1 bool MadDataClassificationWeather::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

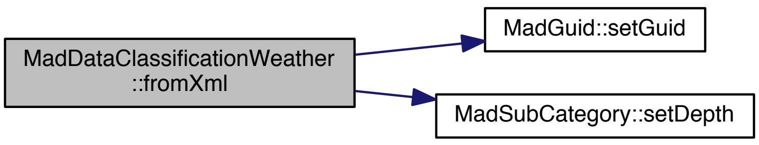
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 173 of file maddataclassificationweather.cpp.

```
{
  QDomDocument myDocument("mydocument");
  myDocument.setContent(theXml);
  QDomElement myTopElement = myDocument.firstChildElement("weather");
  if (myTopElement.isNull())
  {
    // TODO - just make this a warning
    qDebug("the top element couldn't be found!");
    setGuid(myTopElement.attribute("guid"));

    //MadDataClassificationWeather myWeather;
    //QString myPrecipitationXml =
   QString(QDomDocumentFragment().firstChildElement("precipitation").text());
    //myWeather.setPrecipitation(MadSubCategory::fromXml(myPrecipitationXml));

    // the line below works and does the same as the line below it.
    // (QString(myTopElement.firstChildElement("mindata").text() ))=="0" ? mMinData=false : mMinData=true;
    mMinData = QString(myTopElement.firstChildElement("mindata").text()).toInt();
    MadSubCategory myPrecipitationDetails;
    myPrecipitationDetails.setDepth( QString(myTopElement.firstChildElement("precipitation").
  nextSiblingElement("details").nextSiblingElement("depth").text()).toFloat());
    //qDebug()

    /* the following doesn't work
    mPrecipitation = MadUtils::xmlDecode(myTopElement.firstChildElement("precipitation").text()).toInt();
    mTAve = MadUtils::xmlDecode(myTopElement.firstChildElement("tave").text()).toInt();
    mTMin = MadUtils::xmlDecode(myTopElement.firstChildElement("tmin").text()).toFloat();
    mTMax = MadUtils::xmlDecode(myTopElement.firstChildElement("tmax").text()).toInt();
    mRelativeHumidity =
   MadUtils::xmlDecode(myTopElement.firstChildElement("relativehumidity").text()).toInt();
    mWindSpeed = MadUtils::xmlDecode(myTopElement.firstChildElement("windspeed").text()).toInt();
    mGlobalRadiation =
   MadUtils::xmlDecode(myTopElement.firstChildElement("globalradiation").text()).toInt();
    mSunshineHours = MadUtils::xmlDecode(myTopElement.firstChildElement("sunshinehours").text()).toInt();
    mLeafWetness = MadUtils::xmlDecode(myTopElement.firstChildElement("leafwetness").text()).toInt();
    mSoilTemp = MadUtils::xmlDecode(myTopElement.firstChildElement("soiltemp").text()).toInt();
    */
    return true;
  }
  else
    return false;
}
```

Here is the call graph for this function:



#### 6.9.3.2 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

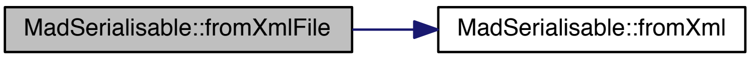
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:

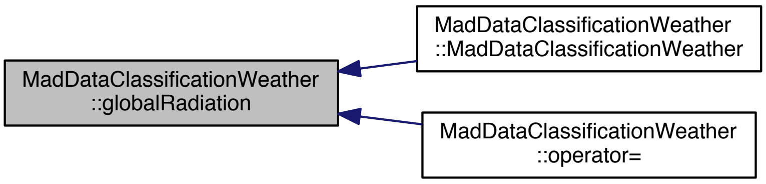


#### 6.9.3.3 MadSubCategory MadDataClassificationWeather::globalRadiation ( ) const

Definition at line 101 of file maddataclassificationweather.cpp.

```
{
  return mGlobalRadiation;
}
```

Here is the caller graph for this function:


MadDataClassificationWeather ::globalRadiation
MadDataClassificationWeather ::MadDataClassificationWeather
MadDataClassificationWeather ::operator=

#### 6.9.3.4 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

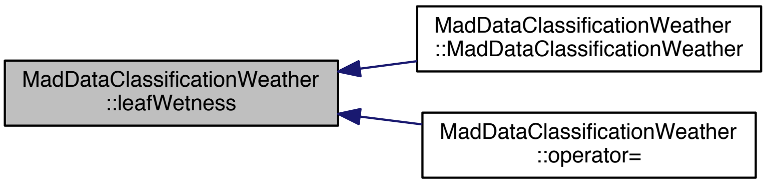
```
{
    return mGuid;
}
```

#### 6.9.3.5 MadSubCategory MadDataClassificationWeather::leafWetness ( ) const

Definition at line 109 of file maddataclassificationweather.cpp.

```
{
  return mLeafWetness;
}
```

Here is the caller graph for this function:


MadDataClassificationWeather ::leafWetness
MadDataClassificationWeather ::MadDataClassificationWeather
MadDataClassificationWeather ::operator=

#### 6.9.3.6 bool MadDataClassificationWeather::minData ( ) const

Definition at line 73 of file maddataclassificationweather.cpp.

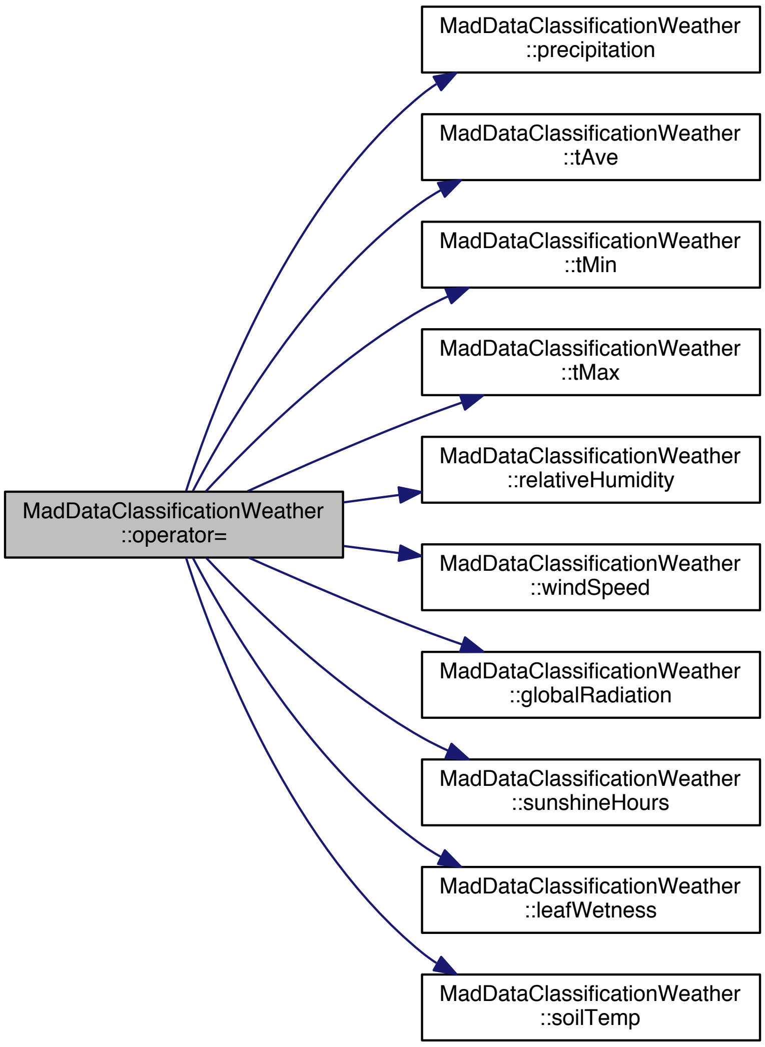
```
{
  return mMinData;
}
```

#### 6.9.3.7 MadDataClassificationWeather & MadDataClassificationWeather::operator= ( const MadDataClassificationWeather & *theData* )

Definition at line 53 of file maddataclassificationweather.cpp.

```
{
  // gracefully handles self assignment
  if (this == &theData) return *this;
  //setGuid(theData.guid());
  mPrecipitation=theData.precipitation();
  mTAve=theData.tAve();
  mTMin=theData.tMin();
  mTMax=theData.tMax();
  mRelativeHumidity=theData.relativeHumidity();
  mWindSpeed=theData.windSpeed();
  mGlobalRadiation=theData.globalRadiation();
  mSunshineHours=theData.sunshineHours();
  mLeafWetness=theData.leafWetness();
  mSoilTemp=theData.soilTemp();

  return *this;
}
```

Here is the call graph for this function:

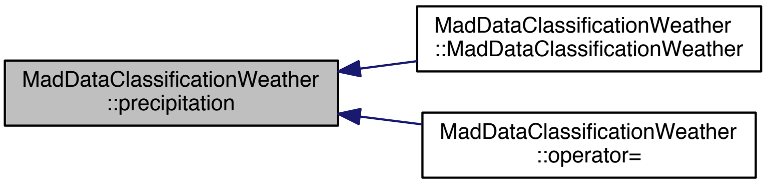

MadDataClassificationWeather ::operator=
MadDataClassificationWeather ::precipitation
MadDataClassificationWeather ::tAve
MadDataClassificationWeather ::tMin
MadDataClassificationWeather ::tMax
MadDataClassificationWeather ::relativeHumidity
MadDataClassificationWeather ::windSpeed
MadDataClassificationWeather ::globalRadiation
MadDataClassificationWeather ::sunshineHours
MadDataClassificationWeather ::leafWetness
MadDataClassificationWeather ::soilTemp

#### 6.9.3.8 MadSubCategory MadDataClassificationWeather::precipitation ( ) const

Definition at line 77 of file maddataclassificationweather.cpp.

```
78 {
79   return mPrecipitation;
80 }
```

Here is the caller graph for this function:

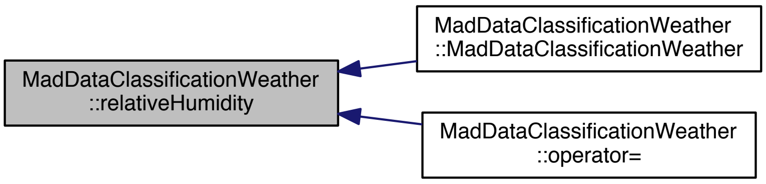

MadDataClassificationWeather ::precipitation
MadDataClassificationWeather ::MadDataClassificationWeather
MadDataClassificationWeather ::operator=

### 6.9.3.9 MadSubCategory MadDataClassificationWeather::relativeHumidity ( ) const

Definition at line 93 of file maddataclassificationweather.cpp.

```
{
  return mRelativeHumidity;
}
```

Here is the caller graph for this function:

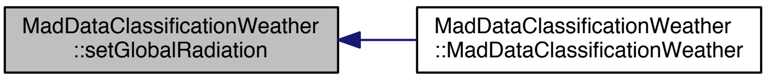

MadDataClassificationWeather ::relativeHumidity
MadDataClassificationWeather ::MadDataClassificationWeather
MadDataClassificationWeather ::operator=

### 6.9.3.10 void MadDataClassificationWeather::setGlobalRadiation ( MadSubCategory *theData* )

Definition at line 153 of file maddataclassificationweather.cpp.

```
{
  mGlobalRadiation = theData;
}
```

Here is the caller graph for this function:


MadDataClassificationWeather ::setGlobalRadiation
MadDataClassificationWeather ::MadDataClassificationWeather

#### 6.9.3.11 void MadGuid::setGuid ( QString *theGuid* = "" ) [inherited]

MadGuid::setGuid.

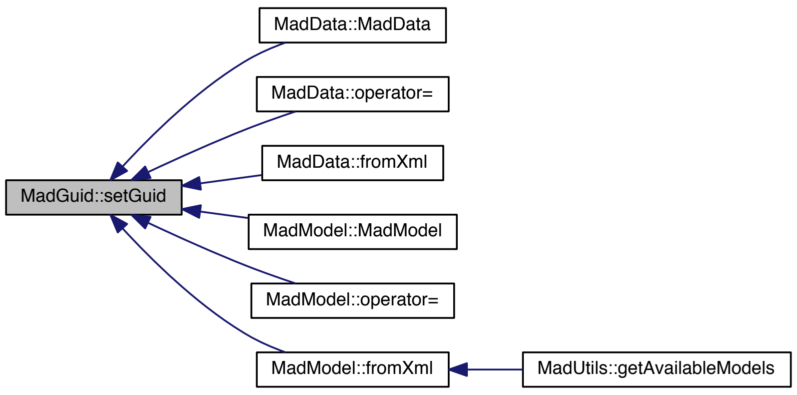
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:

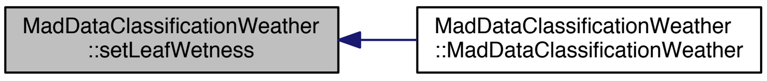


#### 6.9.3.12 void MadDataClassificationWeather::setLeafWetness ( MadSubCategory *theData* )

Definition at line 163 of file maddataclassificationweather.cpp.

```
{
  mLeafWetness = theData;
}
```

Here is the caller graph for this function:


MadDataClassificationWeather ::setLeafWetness
MadDataClassificationWeather ::MadDataClassificationWeather

#### 6.9.3.13 void MadDataClassificationWeather::setMinData ( bool *theBool* )

Definition at line 119 of file maddataclassificationweather.cpp.

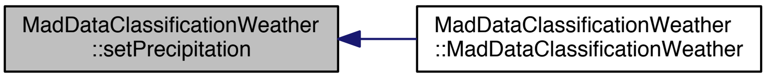
```
{
  mMinData = theBool;
}
```

#### 6.9.3.14 void MadDataClassificationWeather::setPrecipitation ( MadSubCategory *theData* )

Definition at line 123 of file maddataclassificationweather.cpp.

```
{
  mPrecipitation = theData;
}
```

Here is the caller graph for this function:

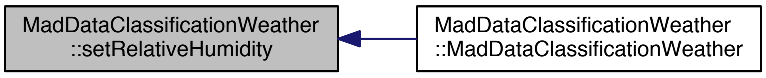

MadDataClassificationWeather ::setPrecipitation
MadDataClassificationWeather ::MadDataClassificationWeather

#### 6.9.3.15 void MadDataClassificationWeather::setRelativeHumidity ( MadSubCategory *theData* )

Definition at line 143 of file maddataclassificationweather.cpp.

```
{
  mRelativeHumidity = theData;
}
```

Here is the caller graph for this function:

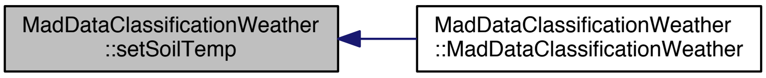

MadDataClassificationWeather ::setRelativeHumidity
MadDataClassificationWeather ::MadDataClassificationWeather

#### 6.9.3.16 void MadDataClassificationWeather::setSoilTemp ( MadSubCategory *theData* )

Definition at line 168 of file maddataclassificationweather.cpp.

```
{
  mSoilTemp = theData;
}
```

Here is the caller graph for this function:

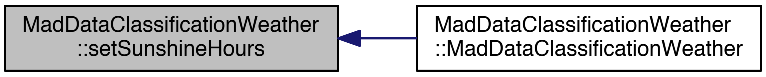

MadDataClassificationWeather ::setSoilTemp
MadDataClassificationWeather ::MadDataClassificationWeather

#### 6.9.3.17 void MadDataClassificationWeather::setSunshineHours ( MadSubCategory *theData* )

Definition at line 158 of file maddataclassificationweather.cpp.

```
{
  mSunshineHours = theData;
}
```

Here is the caller graph for this function:

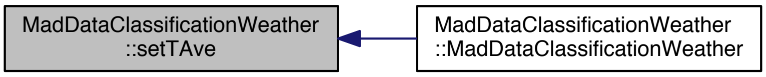

MadDataClassificationWeather ::setSunshineHours
MadDataClassificationWeather ::MadDataClassificationWeather

#### 6.9.3.18 void MadDataClassificationWeather::setTAve ( MadSubCategory *theData* )

Definition at line 128 of file maddataclassificationweather.cpp.

```
{
  mTAve = theData;
}
```

Here is the caller graph for this function:

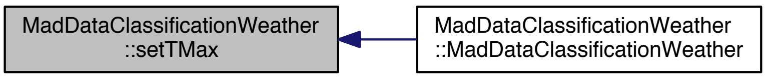

MadDataClassificationWeather ::setTAve
MadDataClassificationWeather ::MadDataClassificationWeather

#### 6.9.3.19 void MadDataClassificationWeather::setTMax ( MadSubCategory *theData* )

Definition at line 138 of file maddataclassificationweather.cpp.

```
{
  mTMax = theData;
}
```

Here is the caller graph for this function:

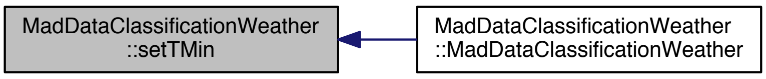

MadDataClassificationWeather ::setTMax
MadDataClassificationWeather ::MadDataClassificationWeather

#### 6.9.3.20 void MadDataClassificationWeather::setTMin ( MadSubCategory *theData* )

Definition at line 133 of file maddataclassificationweather.cpp.

```
{
  mTMin = theData;
}
```

Here is the caller graph for this function:

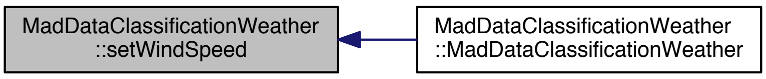

MadDataClassificationWeather ::setTMin
MadDataClassificationWeather ::MadDataClassificationWeather

#### 6.9.3.21 void MadDataClassificationWeather::setWindSpeed ( MadSubCategory *theData* )

Definition at line 148 of file maddataclassificationweather.cpp.

```
{
  mWindSpeed = theData;
}
```

Here is the caller graph for this function:

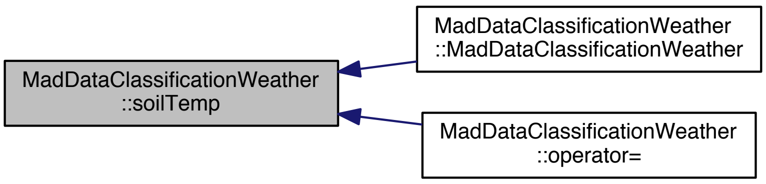

MadDataClassificationWeather ::setWindSpeed
MadDataClassificationWeather ::MadDataClassificationWeather

#### 6.9.3.22 MadSubCategory MadDataClassificationWeather::soilTemp ( ) const

Definition at line 113 of file maddataclassificationweather.cpp.

```
{
  return mSoilTemp;
}
```

Here is the caller graph for this function:

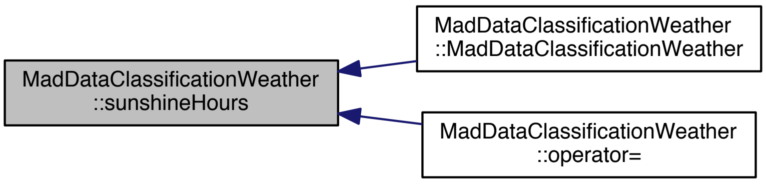

MadDataClassificationWeather ::soilTemp
MadDataClassificationWeather ::MadDataClassificationWeather
MadDataClassificationWeather ::operator=

#### 6.9.3.23 MadSubCategory MadDataClassificationWeather::sunshineHours ( ) const

Definition at line 105 of file maddataclassificationweather.cpp.

```
{
  return mSunshineHours;
}
```

Here is the caller graph for this function:

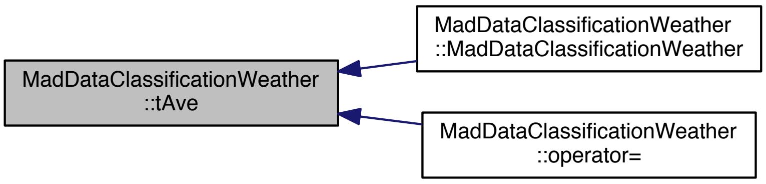

MadDataClassificationWeather ::sunshineHours
MadDataClassificationWeather ::MadDataClassificationWeather
MadDataClassificationWeather ::operator=

#### 6.9.3.24 MadSubCategory MadDataClassificationWeather::tAve ( ) const

Definition at line 81 of file maddataclassificationweather.cpp.

```
{
  return mTAve;
}
```

Here is the caller graph for this function:

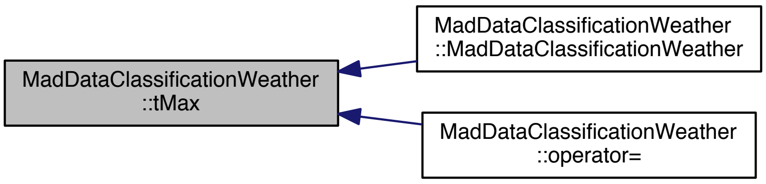

MadDataClassificationWeather ::tAve
MadDataClassificationWeather ::MadDataClassificationWeather
MadDataClassificationWeather ::operator=

#### 6.9.3.25 MadSubCategory MadDataClassificationWeather::tMax ( ) const

Definition at line 89 of file maddataclassificationweather.cpp.

```
{
  return mTMax;
}
```

Here is the caller graph for this function:

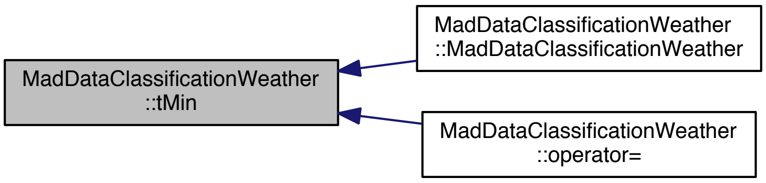

MadDataClassificationWeather ::tMax
MadDataClassificationWeather ::MadDataClassificationWeather
MadDataClassificationWeather ::operator=

#### 6.9.3.26 MadSubCategory MadDataClassificationWeather::tMin ( ) const

Definition at line 85 of file maddataclassificationweather.cpp.

```
{
  return mTMin;
}
```

Here is the caller graph for this function:


MadDataClassificationWeather
::tMin
MadDataClassificationWeather
::MadDataClassificationWeather
MadDataClassificationWeather
::operator=

#### 6.9.3.27 QString MadDataClassificationWeather::toHtml ( )

Return a html text representation of this layer

Definition at line 272 of file maddataclassificationweather.cpp.

```
{
  QString myString;
  //myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  //myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";

  //
  // the following shows example of how to do a couple of things
  //

  //myString+="<tr><td><b>Cals/Kg: </b></td><td>" + QString::number(mCropCalories) + "</td></tr>";
  //QString myCropFodderEnergyType = (mCropFodderEnergyType==0) ? "KCalories" : "TDN";
  //QString myUnits = (mAreaUnits==0) ? "Dunum" : "Hectare";
  //myString+="<tr><td><b>Fodder (kg/" + myUnits + "): </b></td><td>" +
   QString::number(mCropFodderProduction) + "</td></tr>";
  //myString+="<tr><td><b>Fodder Value/Kg: </b></td><td>" + QString::number(mCropFodderValue) +
   "</td></tr>";
  //myString+="<tr><td><b>FodderEnergyType: </b></td><td>" + myCropFodderEnergyType + "</td></tr>";
  //myString+="<tr><td><b>AreaUnits: </b></td><td>" + myUnits + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

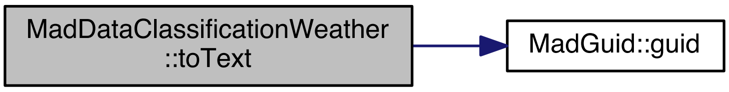
#### 6.9.3.28 QString MadDataClassificationWeather::toText ( )

Return a plain text representation of this layer

Definition at line 263 of file maddataclassificationweather.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  //myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.9.3.29 QString MadDataClassificationWeather::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

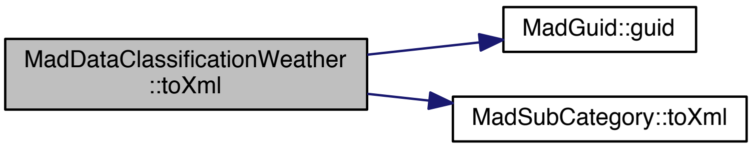
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

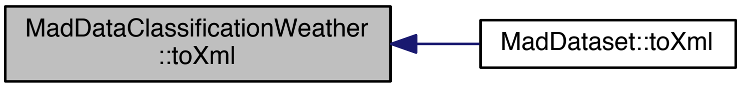
Definition at line 213 of file maddataclassificationweather.cpp.

```
{
  QString myString;
  myString+=QString("  <weather guid=\"" + guid() + "\">\n");

  myString+=QString("    <precipitation>\n");
  myString+=mPrecipitation.toXml();
  myString+=QString("    </precipitation>\n");

  myString+=QString("    <tave>\n");
  myString+=mTAve.toXml();
  myString+=QString("    </tave>\n");

  myString+=QString("    <tmin>\n");
  myString+=mTMin.toXml();
  myString+=QString("    </tmin>\n");

  myString+=QString("    <tmax>\n");
  myString+=mTMax.toXml();
  myString+=QString("    </tmax>\n");

  myString+=QString("    <relativehumidity>\n");
  myString+=mRelativeHumidity.toXml();
  myString+=QString("    </relativehumidity>\n");

  myString+=QString("    <windspeed>\n");
  myString+=mWindSpeed.toXml();
  myString+=QString("    </windspeed>\n");

  myString+=QString("    <globalradiation>\n");
  myString+=mGlobalRadiation.toXml();
  myString+=QString("    </globalradiation>\n");

  myString+=QString("    <sunshinehours>\n");
  myString+=mSunshineHours.toXml();
  myString+=QString("    </sunshinehours>\n");

  myString+=QString("    <leafwetness>\n");
  myString+=mLeafWetness.toXml();
  myString+=QString("    </leafwetness>\n");

  myString+=QString("    <soiltemp>\n");
  myString+=mSoilTemp.toXml();
  myString+=QString("    </soiltemp>\n");

  myString+=QString("  </weather>\n");
  return myString;

}
```

Here is the call graph for this function:



Here is the caller graph for this function:



#### 6.9.3.30 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

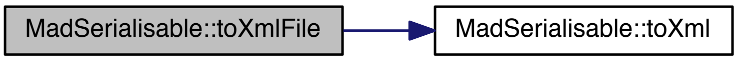
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:

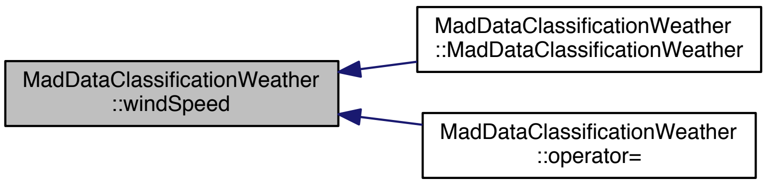


#### 6.9.3.31 MadSubCategory MadDataClassificationWeather::windSpeed ( ) const

Definition at line 97 of file maddataclassificationweather.cpp.

```
{
  return mWindSpeed;
}
```

Here is the caller graph for this function:


MadDataClassificationWeather ::windSpeed
MadDataClassificationWeather ::MadDataClassificationWeather
MadDataClassificationWeather ::operator=

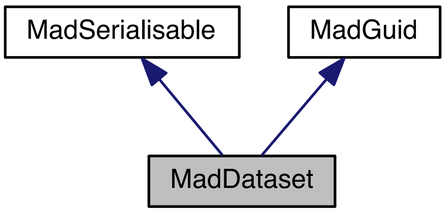
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationweather.-h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationweather.-cpp

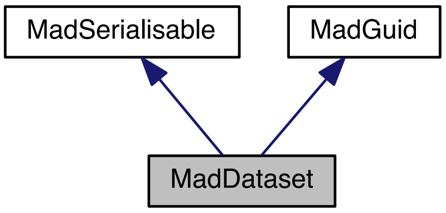
## 6.10 MadDataset Class Reference

```
#include <maddataset.h>
```

Inheritance diagram for MadDataset:



Collaboration diagram for MadDataset:



## Public Member Functions

- MadDataset ()
- MadDataset (const MadDataset &theData)
- MadDataset & operator= (const MadDataset &theDataset)
- QString name () const
- QString description () const
- MadDataClassificationCultivation cultivation () const
- MadDataClassificationInitialValues initialValues () const
- MadDataClassificationPhenology phenology () const
- MadDataClassificationPrevCrop prevCrop () const
- MadDataClassificationSiteData siteData () const
- MadDataClassificationSoil soil () const
- MadDataClassificationWeather weather () const
- MadStateVars stateVars () const
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- void setName (QString theName)
- void setDescription (QString theDescription)
- void setCultivation (MadDataClassificationCultivation theCultivationData)

- void setInitialValues (MadDataClassificationInitialValues theInitialValues)
- void setPhenology (MadDataClassificationPhenology thePhenologyData)
- void setPrevCrop (MadDataClassificationPrevCrop thePrevCropData)
- void setSiteData (MadDataClassificationSiteData theSiteData)
- void setSoil (MadDataClassificationSoil theSoilData)
- void setWeather (MadDataClassificationWeather theWeatherData)
- void setStateVars (MadStateVars theStateVarsData)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

## 6.10.1 Detailed Description

Definition at line 44 of file maddataset.h.

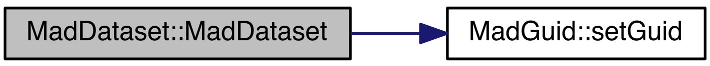
## 6.10.2 Constructor & Destructor Documentation

### 6.10.2.1 MadDataset::MadDataset ( )

Definition at line 34 of file maddataset.cpp.

```
                          : MadSerialisable(), MadGuid()
{
 setGuid();
 mName="No Name Set";
 mDescription="Not Set";
 // we can put in other defaults here, such as
 // mTheme="Valid for all themes";  <-- this doesn't exist though haha
}
```

Here is the call graph for this function:

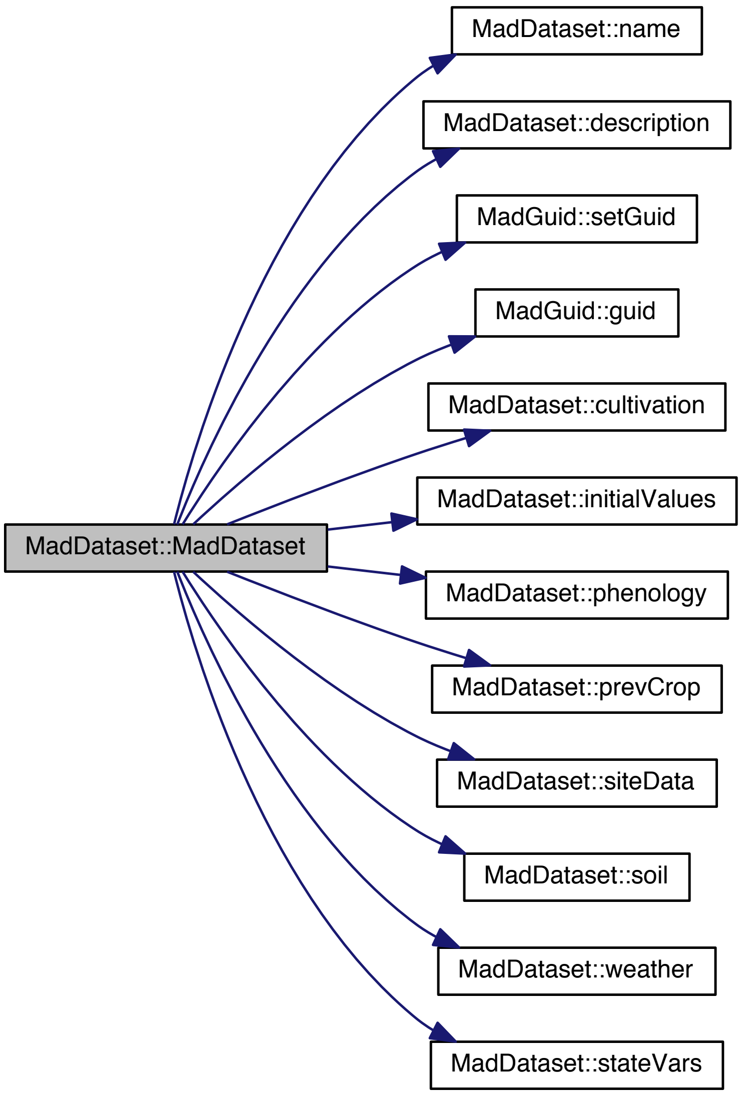


### 6.10.2.2 MadDataset::MadDataset ( const MadDataset & *theData* )

Definition at line 43 of file maddataset.cpp.

```
{
  mName=theData.name();
  mDescription=theData.description();
  setGuid(theData.guid());
  mCultivation=theData.cultivation();
  mInitialValues=theData.initialValues();
  mPhenology=theData.phenology();
  mPrevCrop=theData.prevCrop();
  mSiteData=theData.siteData();
  mSoil=theData.soil();
  mWeather=theData.weather();
  mStateVars=theData.stateVars();
}
```

Here is the call graph for this function:



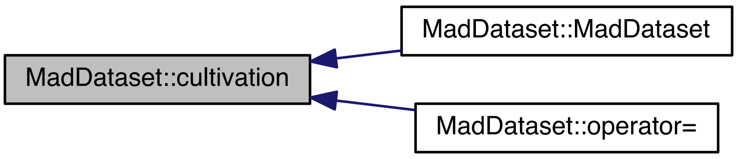
### 6.10.3 Member Function Documentation

#### 6.10.3.1 MadDataClassificationCultivation MadDataset::cultivation ( ) const

Definition at line 86 of file maddataset.cpp.

```
{
  return mCultivation;
}
```

Here is the caller graph for this function:

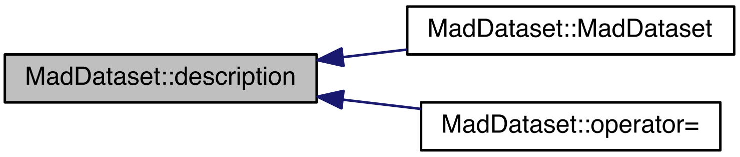


#### 6.10.3.2 QString MadDataset::description ( ) const

Definition at line 81 of file maddataset.cpp.

```
{
  return mDescription;
}
```

Here is the caller graph for this function:



#### 6.10.3.3 bool MadDataset::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

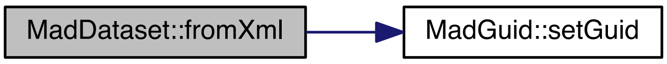
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 171 of file maddataset.cpp.

```
{
    QDomDocument myDocument("mydocument");
    myDocument.setContent(theXml);
    QDomElement myTopElement = myDocument.firstChildElement("model");
    if (myTopElement.isNull())
    {
        //TODO - just make this a warning
        qDebug("the top element couldn't be found!");
        setGuid(myTopElement.attribute("guid"));
        //mName=MadUtils::xmlDecode(myTopElement.firstChildElement("name").text());
        //mDescription=MadUtils::xmlDecode(myTopElement.firstChildElement("description").text());
        //mImageFile=QString(myTopElement.firstChildElement("imageFile").text());
        return true;
    }
    else
    return false;
}
```

Here is the call graph for this function:



### 6.10.3.4 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

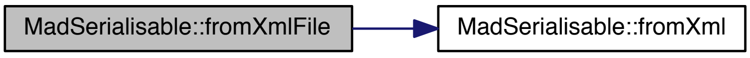
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



### 6.10.3.5 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

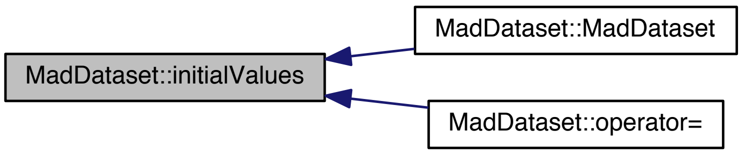
```
{
    return mGuid;
}
```

### 6.10.3.6 MadDataClassificationInitialValues MadDataset::initialValues ( ) const

Definition at line 90 of file maddataset.cpp.

```
{
  return mInitialValues;
}
```

Here is the caller graph for this function:

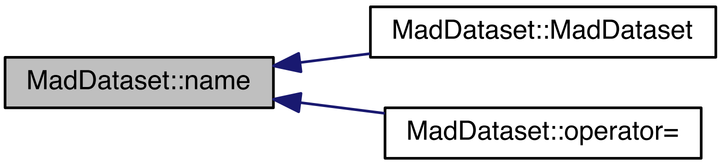


#### 6.10.3.7 QString MadDataset::name ( ) const

Definition at line 77 of file maddataset.cpp.

```
{
  return mName;
}
```

Here is the caller graph for this function:

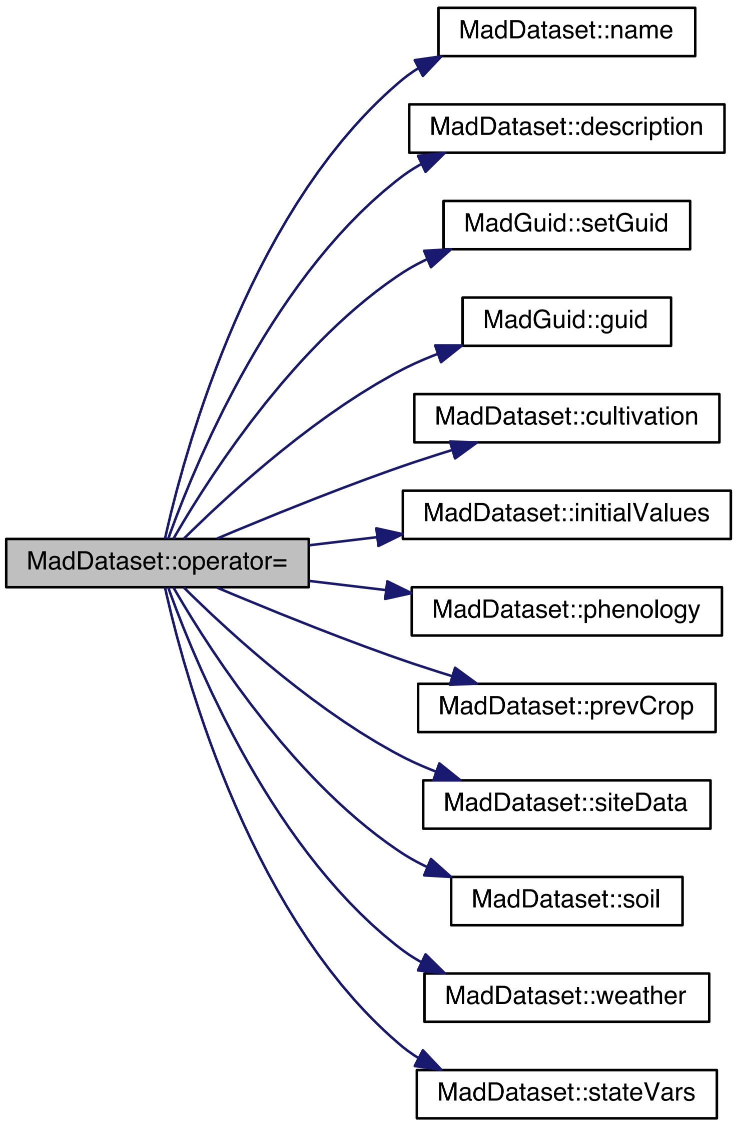


#### 6.10.3.8 MadDataset & MadDataset::operator= ( const MadDataset & *theDataset* )

Definition at line 58 of file maddataset.cpp.

```
{
 // gracefully handles self assignment
 if (this == &theData) return *this;
 mName=theData.name();
 mDescription=theData.description();
 setGuid(theData.guid());
 mCultivation=theData.cultivation();
 mInitialValues=theData.initialValues();
 mPhenology=theData.phenology();
 mPrevCrop=theData.prevCrop();
 mSiteData=theData.siteData();
 mSoil=theData.soil();
 mWeather=theData.weather();
 mStateVars=theData.stateVars();
 return *this;
}
```

Here is the call graph for this function:

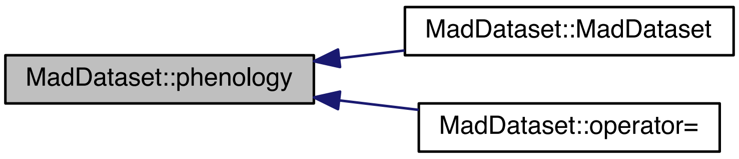


### 6.10.3.9 MadDataClassificationPhenology MadDataset::phenology ( ) const

Definition at line 94 of file maddataset.cpp.

```
{
  return mPhenology;
}
```

Here is the caller graph for this function:

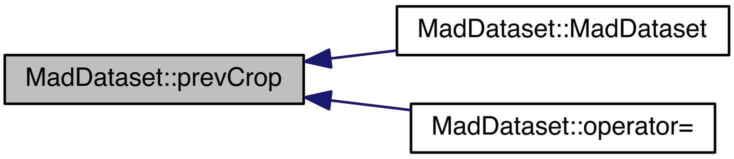


#### 6.10.3.10 **MadDataClassificationPrevCrop** MadDataset::prevCrop ( ) const

Definition at line 98 of file maddataset.cpp.

```
{
  return mPrevCrop;
}
```

Here is the caller graph for this function:



#### 6.10.3.11 void MadDataset::setCultivation ( **MadDataClassificationCultivation** *theCultivationData* )

Definition at line 131 of file maddataset.cpp.

```
{
  mCultivation = theCultivationData;
}
```

#### 6.10.3.12 void MadDataset::setDescription ( QString *theDescription* )

Definition at line 126 of file maddataset.cpp.

```
{
  mDescription = theDescription;
}
```

#### 6.10.3.13 void MadGuid::setGuid ( QString *theGuid* = "" ) [inherited]

MadGuid::setGuid.

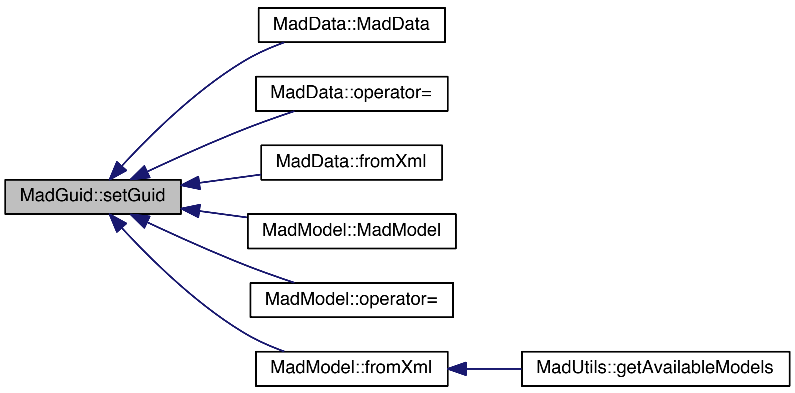
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:



#### 6.10.3.14 void MadDataset::setInitialValues ( MadDataClassificationInitialValues *theInitialValues* )

Definition at line 136 of file maddataset.cpp.

```
{
  mInitialValues = theInitialValues;
}
```

#### 6.10.3.15 void MadDataset::setName ( QString *theName* )

Definition at line 121 of file maddataset.cpp.

```
{
  mName = theName;
}
```

#### 6.10.3.16 void MadDataset::setPhenology ( MadDataClassificationPhenology *thePhenologyData* )

Definition at line 141 of file maddataset.cpp.

```
{
  mPhenology = thePhenologyData;
}
```

#### 6.10.3.17 void MadDataset::setPrevCrop ( MadDataClassificationPrevCrop *thePrevCropData* )

Definition at line 146 of file maddataset.cpp.

```
{
  mPrevCrop = thePrevCropData;
}
```

#### 6.10.3.18 void MadDataset::setSiteData ( MadDataClassificationSiteData *theSiteData* )

Definition at line 151 of file maddataset.cpp.

```
{
  mSiteData = theSiteData;
}
```

#### 6.10.3.19 void MadDataset::setSoil ( MadDataClassificationSoil *theSoilData* )

Definition at line 156 of file maddataset.cpp.

```
{
  mSoil = theSoilData;
}
```

#### 6.10.3.20 void MadDataset::setStateVars ( MadStateVars *theStateVarsData* )

Definition at line 166 of file maddataset.cpp.

```
{
  mStateVars = theStateVarsData;
}
```

#### 6.10.3.21 void MadDataset::setWeather ( MadDataClassificationWeather *theWeatherData* )

Definition at line 161 of file maddataset.cpp.

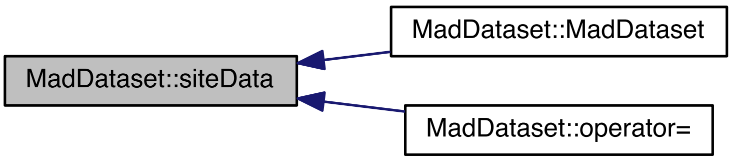
```
{
  mWeather = theWeatherData;
}
```

#### 6.10.3.22 MadDataClassificationSiteData MadDataset::siteData ( ) const

Definition at line 102 of file maddataset.cpp.

```
{
  return mSiteData;
}
```

Here is the caller graph for this function:

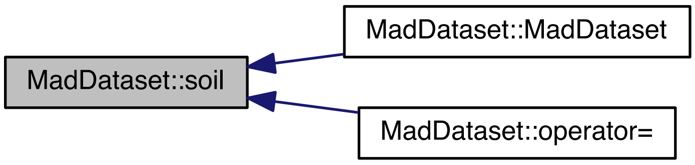


#### 6.10.3.23 MadDataClassificationSoil MadDataset::soil ( ) const

Definition at line 106 of file maddataset.cpp.

```
{
  return mSoil;
}
```

Here is the caller graph for this function:

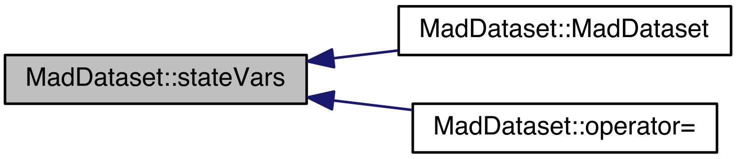


#### 6.10.3.24 MadStateVars MadDataset::stateVars ( ) const

Definition at line 114 of file maddataset.cpp.

```
{
  return mStateVars;
}
```

Here is the caller graph for this function:



### 6.10.3.25 QString MadDataset::toHtml ( )

Return a html text representation of this layer

Definition at line 217 of file maddataset.cpp.

```
{
  QString myString;
  //myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  //myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";

  //
  // the following shows example of how to do a couple of things
  //

  //myString+="<tr><td><b>Cals/Kg: </b></td><td>" + QString::number(mCropCalories) + "</td></tr>";
  //QString myCropFodderEnergyType = (mCropFodderEnergyType==0) ? "KCalories" : "TDN";
  //QString myUnits = (mAreaUnits==0) ? "Dunum" : "Hectare";
  //myString+="<tr><td><b>Fodder (kg/" + myUnits + "): </b></td><td>" +
   QString::number(mCropFodderProduction) + "</td></tr>";
  //myString+="<tr><td><b>Fodder Value/Kg: </b></td><td>" + QString::number(mCropFodderValue) +
   "</td></tr>";
  //myString+="<tr><td><b>FodderEnergyType: </b></td><td>" + myCropFodderEnergyType + "</td></tr>";
  //myString+="<tr><td><b>AreaUnits: </b></td><td>" + myUnits + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

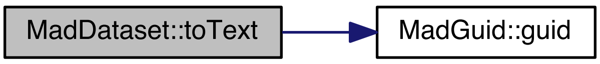
### 6.10.3.26 QString MadDataset::toText ( )

Return a plain text representation of this layer

Definition at line 208 of file maddataset.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  //myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



### 6.10.3.27 QString MadDataset::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

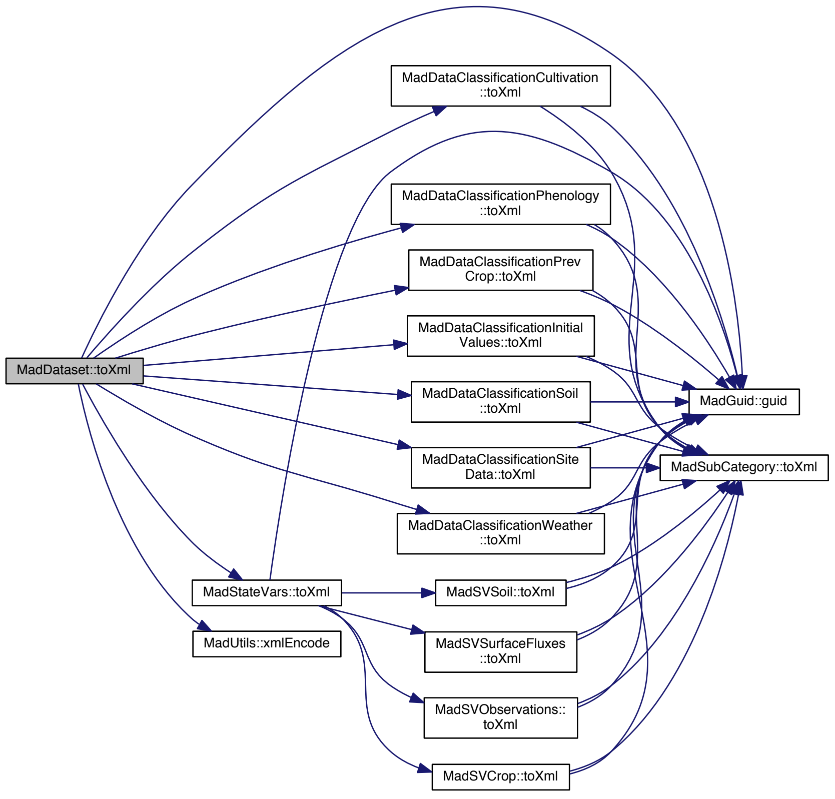
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 190 of file maddataset.cpp.

```
{
  QString myString;
  myString+=QString("<dataset guid=\"" + guid() + "\">\n");
  myString+=QString("  <name>" + MadUtils::xmlEncode(mName) + "</name>\n");
  myString+=QString("  <description>" + MadUtils::xmlEncode(mDescription) + "
  </description>\n");
  myString+=mCultivation.toXml();
  myString+=mPhenology.toXml();
  myString+=mPrevCrop.toXml();
  myString+=mInitialValues.toXml();
  myString+=mSoil.toXml();
  myString+=mSiteData.toXml();
  myString+=mWeather.toXml();
  myString+=mStateVars.toXml();
  myString+=QString("</dataset>\n");
  return myString;
}
```

Here is the call graph for this function:


MadDataset::toXml
MadDataClassificationCultivation ::toXml
MadDataClassificationPhenology ::toXml
MadDataClassificationPrev Crop::toXml
MadDataClassificationInitial Values::toXml
MadDataClassificationSoil ::toXml
MadDataClassificationSite Data::toXml
MadDataClassificationWeather ::toXml
MadStateVars::toXml
MadUtils::xmlEncode
MadSVSoil::toXml
MadSVSurfaceFluxes ::toXml
MadSVObservations:: toXml
MadSVCrop::toXml
MadGuid::guid
MadSubCategory::toXml

### 6.10.3.28 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) `[virtual], [inherited]`

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

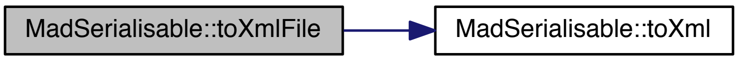
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:

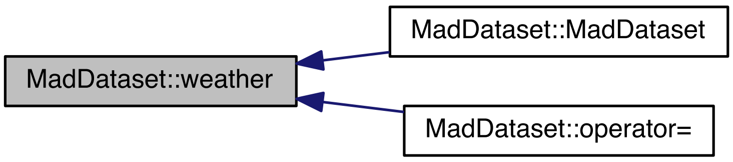


#### 6.10.3.29 MadDataClassificationWeather MadDataset::weather ( ) const

Definition at line 110 of file maddataset.cpp.

```
{
  return mWeather;
}
```

Here is the caller graph for this function:



The documentation for this class was generated from the following files:

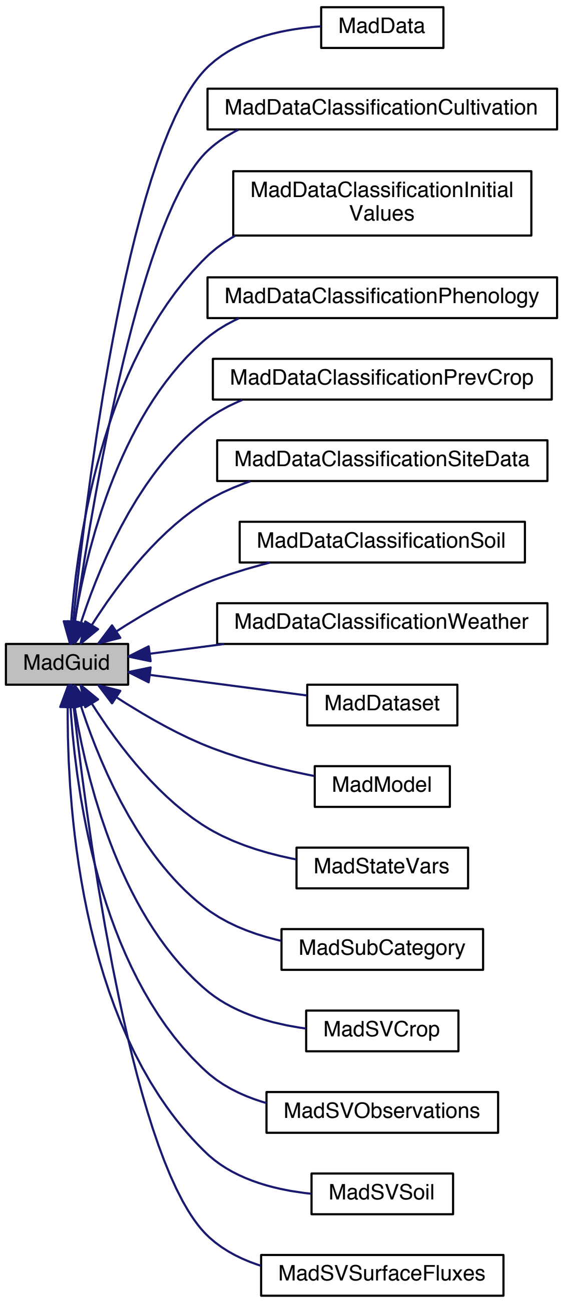
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddataset.h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddataset.cpp

## 6.11 MadGuid Class Reference

The MadGuid class An abstract base class that has a Globally Unique Identifier (GUID) to represent a unique instance.

`#include <madguid.h>`

Inheritance diagram for MadGuid:


MadData
MadDataClassificationCultivation
MadDataClassificationInitial Values
MadDataClassificationPhenology
MadDataClassificationPrevCrop
MadDataClassificationSiteData
MadDataClassificationSoil
MadDataClassificationWeather
MadGuid
MadDataset
MadModel
MadStateVars
MadSubCategory
MadSVCrop
MadSVObservations
MadSVSoil
MadSVSurfaceFluxes

## Public Member Functions

- MadGuid ()
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.11.1 Detailed Description

The MadGuid class An abstract base class that has a Globally Unique Identifier (GUID) to represent a unique instance.

Definition at line 32 of file madguid.h.

### 6.11.2 Constructor & Destructor Documentation

#### 6.11.2.1 MadGuid::MadGuid ( )

Constructor

Definition at line 28 of file madguid.cpp.

```
{
}
```

### 6.11.3 Member Function Documentation

#### 6.11.3.1 QString MadGuid::guid ( ) const

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

```
{
    return mGuid;
}
```

#### 6.11.3.2 void MadGuid::setGuid ( QString *theGuid* = " " )

MadGuid::setGuid.

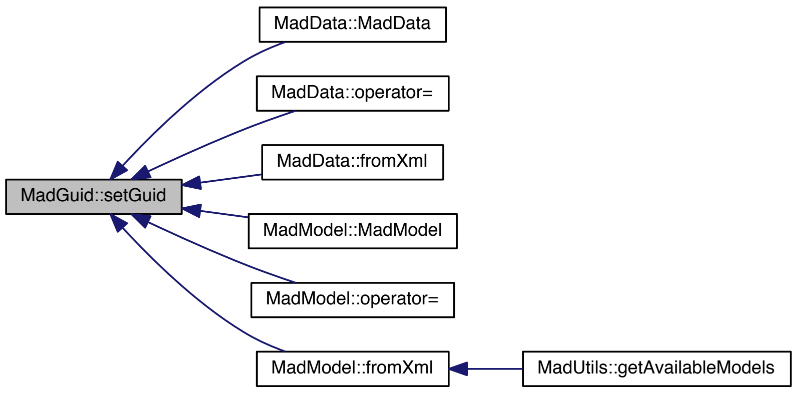
**Parameters**

|  |  |
|---|---|
| *theGuid* |  |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:



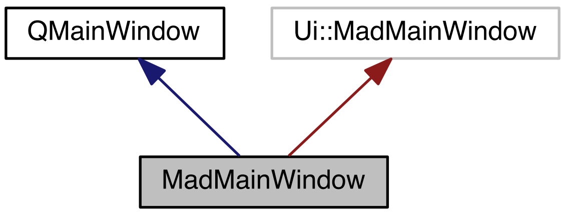
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madguid.h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madguid.cpp

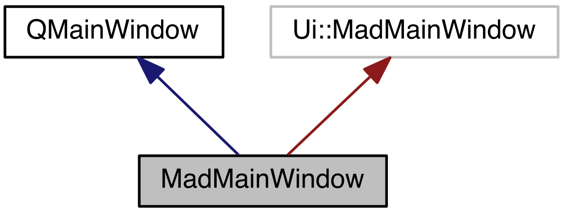
## 6.12 MadMainWindow Class Reference

`#include <madmainwindow.h>`

Inheritance diagram for MadMainWindow:



Collaboration diagram for MadMainWindow:



### Public Member Functions

- MadMainWindow (QWidget ∗parent=0)
- QString modelText () const
- void setModelText (QString theModelText)

### Protected Member Functions

- void changeEvent (QEvent ∗e)

  *changeEvent for translations in the future*

## 6.12.1 Detailed Description

This is the main GUI class

**Author**

Jason Jorgenson

Definition at line 44 of file madmainwindow.h.

## 6.12.2 Constructor & Destructor Documentation

### 6.12.2.1 MadMainWindow::MadMainWindow ( QWidget ∗ *parent* = 0 ) [explicit]

This is the main form GUI of MAD (Macsur ADapter) It sets up the required slot connections and initialises the GUI

**Parameters**

| | |
|---|---|
| *parent* | |

Definition at line 35 of file madmainwindow.cpp.

```
                                          :
    QMainWindow(parent)
{
    setupUi(this);
    // the key to making the revision autoupdate is to use a feature in svn
    // that will update keywords on commits.  to make this work, you need to:
    //   svn propset svn:keywords "Revision" madmainwindow.cpp
    // and then it works!  note that you need to 'touch' madmainform.cpp every
    // commit for this to work.  This could be simply adding/removing a LF in
    // this file (madmainform.cpp)
    lblVersion->setText(QString("Version: %1").arg(VERSION)+ " "
                        + QString("$Revision: 141 $").replace("$",""));
}
```

### 6.12.3 Member Function Documentation

#### 6.12.3.1 void MadMainWindow::changeEvent ( QEvent ∗ *e* ) [protected]

changeEvent for translations in the future

**Parameters**

| | |
|---|---|
| *e* | |

Definition at line 59 of file madmainwindow.cpp.

```
{
    QMainWindow::changeEvent(e);
    switch (e->type()) {
    case QEvent::LanguageChange:
        retranslateUi(this);
        break;
    default:
        break;
    }
}
```

#### 6.12.3.2 QString MadMainWindow::modelText ( ) const

Definition at line 49 of file madmainwindow.cpp.

```
{
  return mModelText;
}
```

#### 6.12.3.3 void MadMainWindow::setModelText ( QString *theModelText* )

Definition at line 54 of file madmainwindow.cpp.

```
{
  mModelText=theModelText;
}
```

The documentation for this class was generated from the following files:

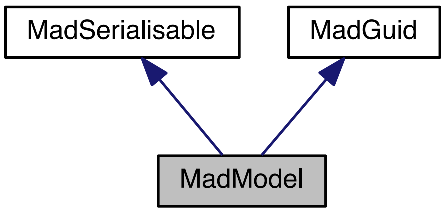
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/madmainwindow.h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/madmainwindow.cpp

## 6.13 MadModel Class Reference

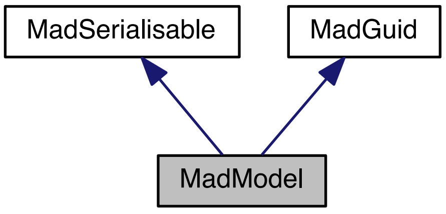
The MadModel class, to represent a ModelTheme.

`#include <madmodel.h>`

Inheritance diagram for MadModel:



Collaboration diagram for MadModel:



### Public Member Functions

- MadModel ()
- MadModel (const MadModel &theModel)
- MadModel & operator= (const MadModel &theModel)
- QString name () const
- QString description () const
- QString imageFile () const
- void setName (QString theName)
- void setDescription (QString theDescription)
- void setImageFile (QString theImageFileName)
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

*MadGuid::guid.*

- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.13.1 Detailed Description

The MadModel class, to represent a ModelTheme.

Definition at line 56 of file madmodel.h.

### 6.13.2 Constructor & Destructor Documentation

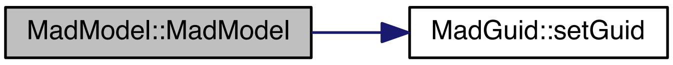
#### 6.13.2.1 MadModel::MadModel ( )

Constructor .

Definition at line 33 of file madmodel.cpp.

```
                     : MadSerialisable(), MadGuid()
{
    setGuid();
    mName="No Name Set";
    mDescription="Not Set";
}
```

Here is the call graph for this function:



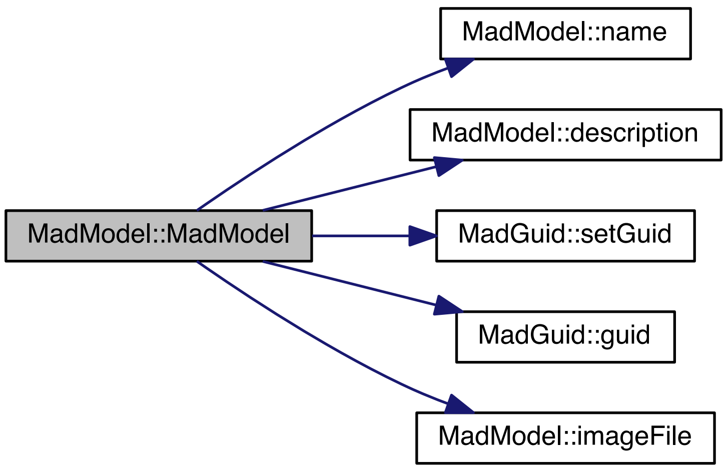
#### 6.13.2.2 MadModel::MadModel ( const **MadModel** & *theModel* )

Destructor . copy constructor

Definition at line 46 of file madmodel.cpp.

```
{
    mName=theModel.name();
    mDescription=theModel.description();
    setGuid(theModel.guid());
    mImageFile=theModel.imageFile();
}
```

Here is the call graph for this function:



### 6.13.3 Member Function Documentation

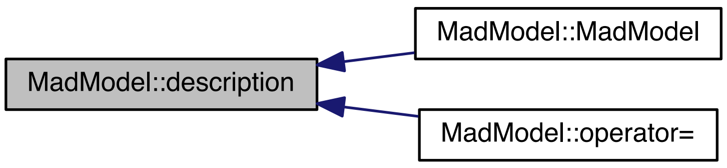
#### 6.13.3.1 QString MadModel::description ( ) const

The description of this model

Definition at line 70 of file madmodel.cpp.

```
{
    return mDescription;
}
```

Here is the caller graph for this function:



#### 6.13.3.2 bool MadModel::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

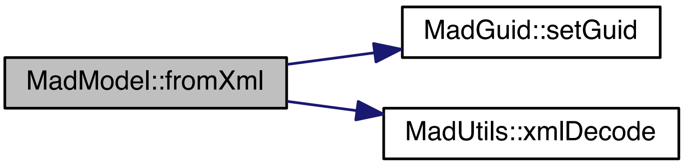
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

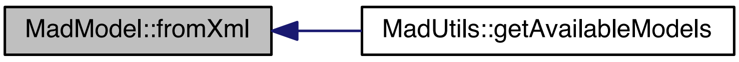
Definition at line 97 of file madmodel.cpp.

```
{
    QDomDocument myDocument("mydocument");
     myDocument.setContent(theXml);
     QDomElement myTopElement = myDocument.firstChildElement("model");
     if (myTopElement.isNull())
     {
         //TODO - just make this a warning
         qDebug("the top element couldn't be found!");
         setGuid(myTopElement.attribute("guid"));
         mName=MadUtils::xmlDecode(myTopElement.firstChildElement("name").text());
         mDescription=MadUtils::xmlDecode(myTopElement.firstChildElement("description").
   text());
         mImageFile=QString(myTopElement.firstChildElement("imageFile").text());
         return true;
     }
     else
     return false;
}
```

Here is the call graph for this function:



Here is the caller graph for this function:



### 6.13.3.3 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| *theFileName* | |
|---|---|

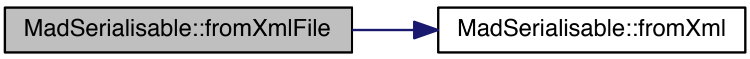
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



#### 6.13.3.4 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

```
{
    return mGuid;
}
```

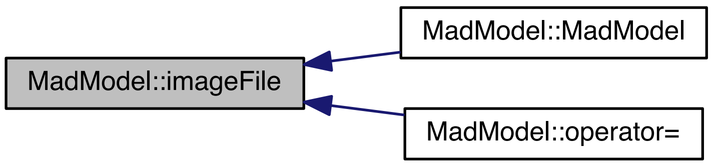
#### 6.13.3.5 QString MadModel::imageFile ( ) const

The image file associated with the model

Definition at line 75 of file madmodel.cpp.

```
{
    return mImageFile;
}
```

Here is the caller graph for this function:



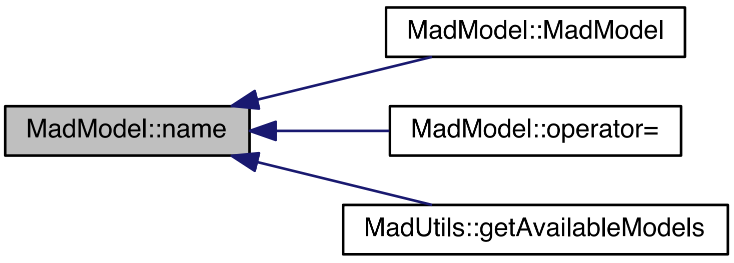
#### 6.13.3.6 QString MadModel::name ( ) const

The name of this model

Definition at line 65 of file madmodel.cpp.

```
{
    return mName;
}
```

Here is the caller graph for this function:



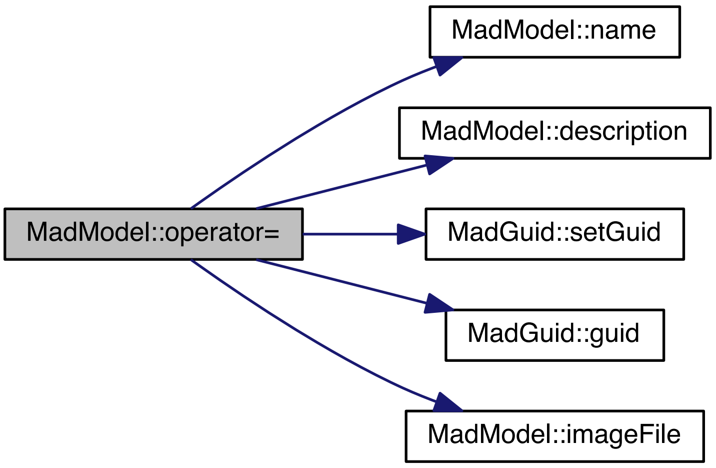
#### 6.13.3.7 MadModel & MadModel::operator= ( const MadModel & *theModel* )

Assignement operator

Definition at line 54 of file madmodel.cpp.

```
{
    if (this == &theModel) return *this; // gracefully handles self assignment

    mName=theModel.name();
    mDescription=theModel.description();
    setGuid(theModel.guid());
    mImageFile=theModel.imageFile();
    return *this;
}
```

Here is the call graph for this function:



#### 6.13.3.8 void MadModel::setDescription ( QString *theDescription* )

Set the model description

**See Also**

description()

Definition at line 87 of file madmodel.cpp.

```
{
    mDescription=theDescription;
}
```

#### 6.13.3.9 void MadGuid::setGuid ( QString *theGuid* = " " ) [inherited]

MadGuid::setGuid.

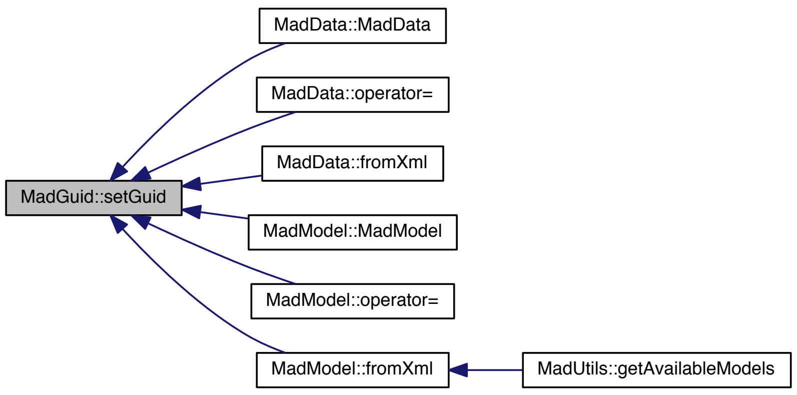
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:



#### 6.13.3.10 void MadModel::setImageFile ( QString *theImageFileName* )

Set the image file

**See Also**

imageFile()

Definition at line 92 of file madmodel.cpp.

```
{
    mImageFile=theImageFileName;
}
```

#### 6.13.3.11 void MadModel::setName ( QString *theName* )

Set the modelName

**See Also**

name()

Definition at line 82 of file madmodel.cpp.

```
{
    mName=theName;
}
```

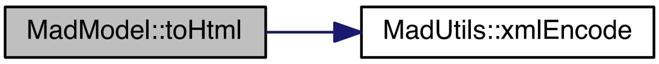
#### 6.13.3.12 QString MadModel::toHtml ( )

Return a html text representation of this layer

Definition at line 147 of file madmodel.cpp.

```
{
  QString myString;
  myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";

  //
  // the following shows example of how to do a couple of things
  //

  //myString+="<tr><td><b>Cals/Kg: </b></td><td>" + QString::number(mCropCalories) + "</td></tr>";
  //QString myCropFodderEnergyType = (mCropFodderEnergyType==0) ? "KCalories" : "TDN";
  //QString myUnits = (mAreaUnits==0) ? "Dunum" : "Hectare";
  //myString+="<tr><td><b>Fodder (kg/" + myUnits + "): </b></td><td>" +
   QString::number(mCropFodderProduction) + "</td></tr>";
  //myString+="<tr><td><b>Fodder Value/Kg: </b></td><td>" + QString::number(mCropFodderValue) +
   "</td></tr>";
  //myString+="<tr><td><b>FodderEnergyType: </b></td><td>" + myCropFodderEnergyType + "</td></tr>";
  //myString+="<tr><td><b>AreaUnits: </b></td><td>" + myUnits + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

Here is the call graph for this function:



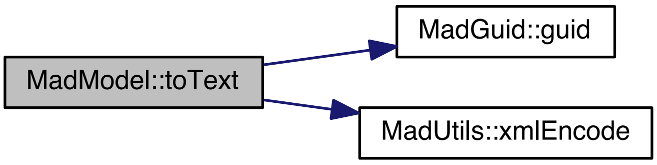
#### 6.13.3.13 QString MadModel::toText ( )

Return a plain text representation of this layer

Definition at line 138 of file madmodel.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.13.3.14 QString MadModel::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

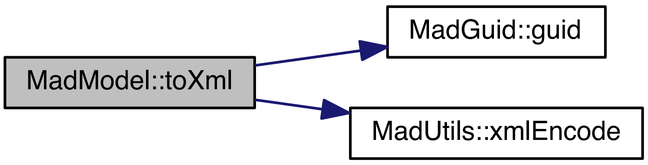
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 116 of file madmodel.cpp.

```
{
  QString myString;
  myString+=QString("<model guid=\"" + guid() + "\">\n");
    myString+=QString("  <name>" + MadUtils::xmlEncode(mName) + "</name>\n");
  myString+=QString("  <description>" + MadUtils::xmlEncode(mDescription) + "
  </description>\n");


//  switch (mAreaUnits)
//  {
//    case Dunum:
//      myString+=QString("  <areaUnits>Dunum</areaUnits>\n");
//      break;
//    case Hectare:
//      myString+=QString("  <areaUnits>Hectare</areaUnits>\n");
//      break;
//  }
  myString+=QString("  <imageFile>" + MadUtils::xmlEncode(mImageFile) + "</imageFile>\n"
  );
  myString+=QString("</model>\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.13.3.15 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

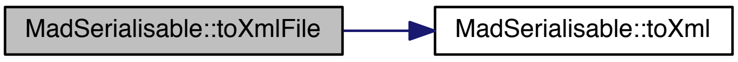
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



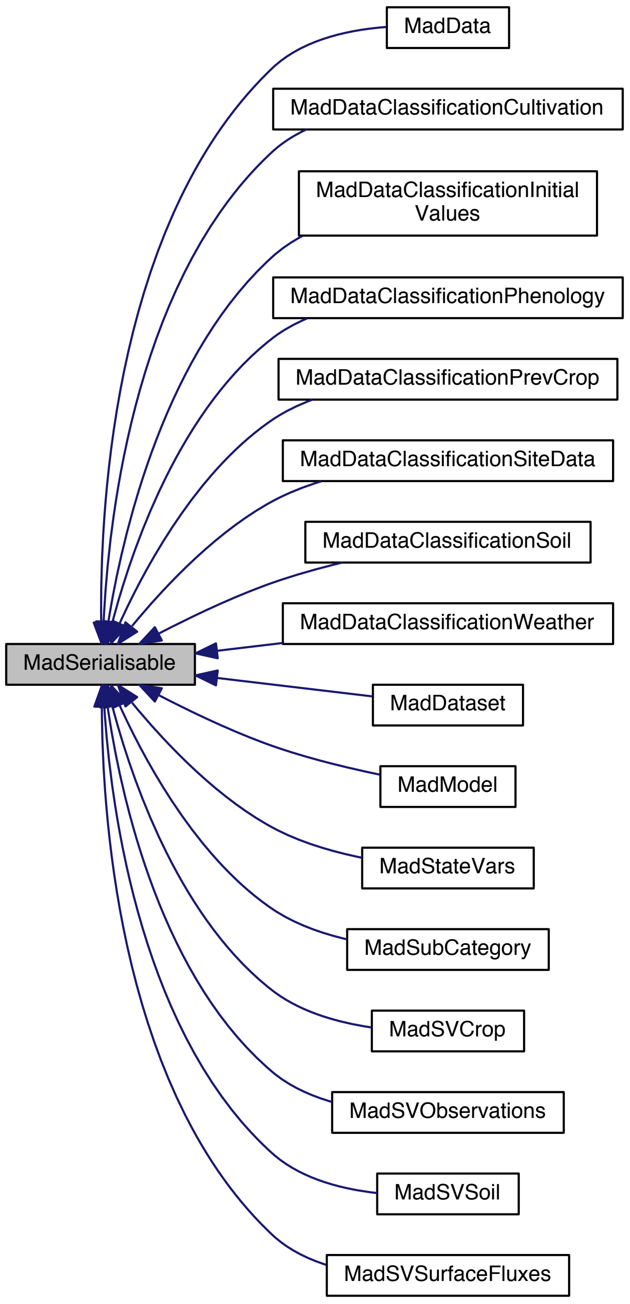
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madmodel.h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madmodel.cpp

## 6.14 MadSerialisable Class Reference

`#include <madserialisable.h>`

Inheritance diagram for MadSerialisable:


MadData
MadDataClassificationCultivation
MadDataClassificationInitial Values
MadDataClassificationPhenology
MadDataClassificationPrevCrop
MadDataClassificationSiteData
MadDataClassificationSoil
MadDataClassificationWeather
MadSerialisable
MadDataset
MadModel
MadStateVars
MadSubCategory
MadSVCrop
MadSVObservations
MadSVSoil
MadSVSurfaceFluxes

## Public Member Functions

- MadSerialisable ()

  *MadSerialisable Constructor.*
- virtual QString toXml ()=0

  *toXml Write this object to xml and return result as qstring (virtual)*
- virtual bool toXmlFile (const QString theFileName)

*toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*

- virtual bool fromXml (const QString theXml)=0

  *fromXml Read this object from xml*

- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*

## 6.14.1 Detailed Description

An abstract base class for any class that is serialiseable to xml

**Author**

Tim Sutton, Jason Jorgenson

Definition at line 50 of file madserialisable.h.

## 6.14.2 Constructor & Destructor Documentation

### 6.14.2.1 MadSerialisable::MadSerialisable ( )

MadSerialisable Constructor.

Definition at line 49 of file madserialisable.cpp.

```
50 {
51 }
```

## 6.14.3 Member Function Documentation

### 6.14.3.1 virtual bool MadSerialisable::fromXml ( const QString *theXml* ) [pure virtual]

fromXml Read this object from xml

**Parameters**

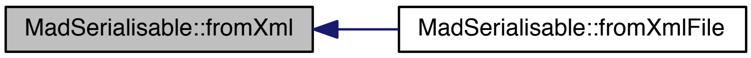
| | |
|---|---|
| *theXml* | |

**Returns**

result as true for success, false for failure (virtual)

Implemented in MadModel, MadData, MadSVCrop, MadSVSurfaceFluxes, MadDataset, MadStateVars, MadDataClassificationWeather, MadDataClassificationCultivation, MadDataClassificationSoil, MadDataClassificationPrevCrop, MadSVSoil, MadDataClassificationSiteData, MadSVObservations, MadDataClassificationInitialValues, MadSubCategory, and MadDataClassificationPhenology.

Here is the caller graph for this function:



#### 6.14.3.2 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

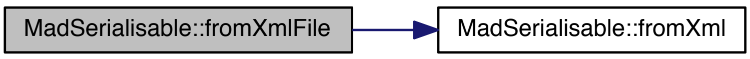
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



#### 6.14.3.3 virtual QString MadSerialisable::toXml ( ) [pure virtual]

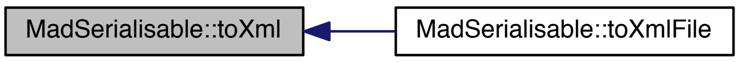
toXml Write this object to xml and return result as qstring (virtual)

Desctructor .

**Returns**

Implemented in MadModel, MadData, MadSVCrop, MadSVSurfaceFluxes, MadDataset, MadStateVars, MadDataClassificationWeather, MadDataClassificationCultivation, MadDataClassificationSoil, MadDataClassificationPrevCrop, MadSVSoil, MadDataClassificationSiteData, MadSVObservations, MadDataClassificationInitialValues, MadSubCategory, and MadDataClassificationPhenology.

Here is the caller graph for this function:



#### 6.14.3.4 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

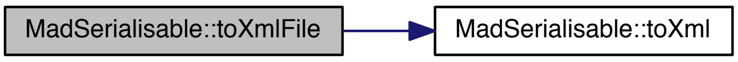
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



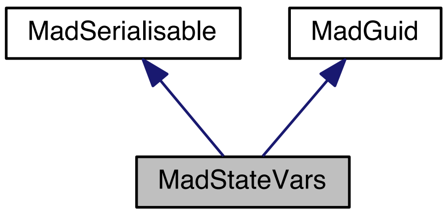
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madserialisable.h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madserialisable.cpp

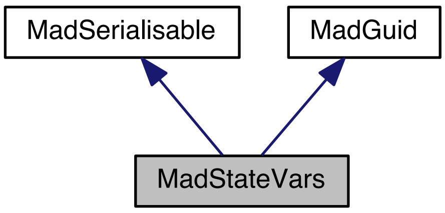
## 6.15 MadStateVars Class Reference

`#include <madstatevars.h>`

Inheritance diagram for MadStateVars:



Collaboration diagram for MadStateVars:



### Public Member Functions

- MadStateVars ()
- MadStateVars (const MadStateVars &theData)
- MadStateVars & operator= (const MadStateVars &theData)
- MadSVCrop cropCategories () const
- MadSVSoil soilCategories () const
- MadSVSurfaceFluxes surfaceFluxesCategories () const
- MadSVObservations observationCategories () const
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- void setCropCategories (MadSVCrop theMadSVCrop)
- void setSoilCategories (MadSVSoil theData)
- void setSurfaceFluxesCategories (MadSVSurfaceFluxes theData)
- void setObservationCategories (MadSVObservations theData)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.15.1 Detailed Description

Definition at line 43 of file madstatevars.h.

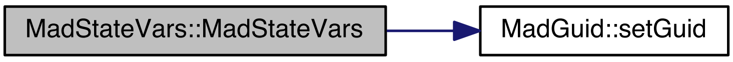
## 6.15.2 Constructor & Destructor Documentation

### 6.15.2.1 MadStateVars::MadStateVars ( )

Definition at line 36 of file madstatevars.cpp.

```
                              : MadSerialisable(), MadGuid()
{
  setGuid();
}
```

Here is the call graph for this function:

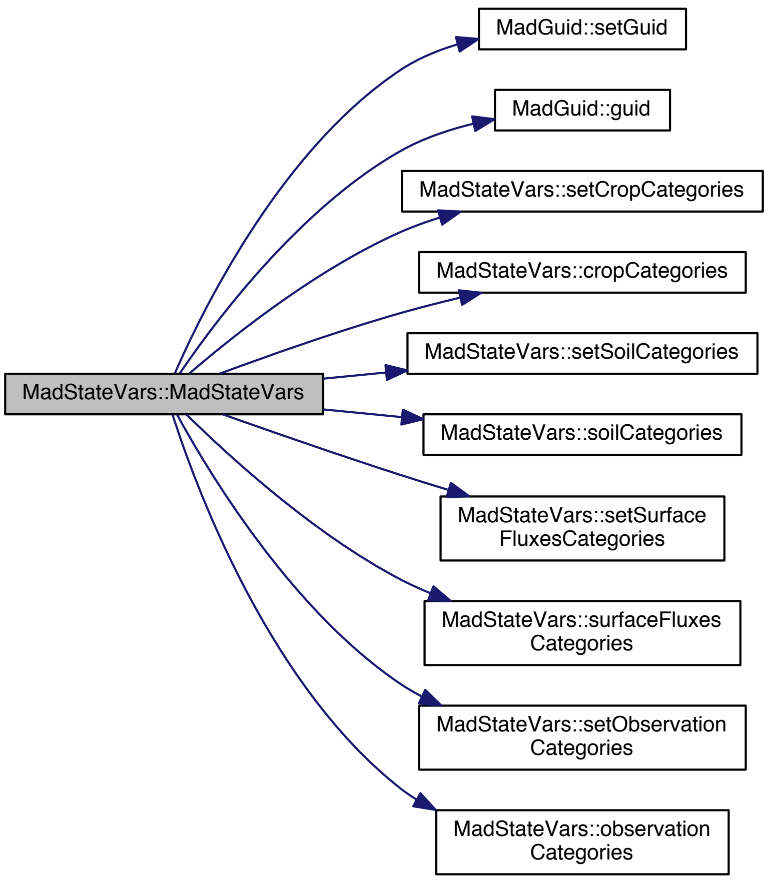


### 6.15.2.2 MadStateVars::MadStateVars ( const **MadStateVars** & *theData* )

Definition at line 41 of file madstatevars.cpp.

```
{
  setGuid(theData.guid());
  setCropCategories(theData.cropCategories());
  setSoilCategories(theData.soilCategories());
  setSurfaceFluxesCategories(theData.
   surfaceFluxesCategories());
  setObservationCategories(theData.observationCategories());
}
```

Here is the call graph for this function:


MadGuid::setGuid
MadGuid::guid
MadStateVars::setCropCategories
MadStateVars::cropCategories
MadStateVars::setSoilCategories
MadStateVars::MadStateVars
MadStateVars::soilCategories
MadStateVars::setSurface FluxesCategories
MadStateVars::surfaceFluxes Categories
MadStateVars::setObservation Categories
MadStateVars::observation Categories

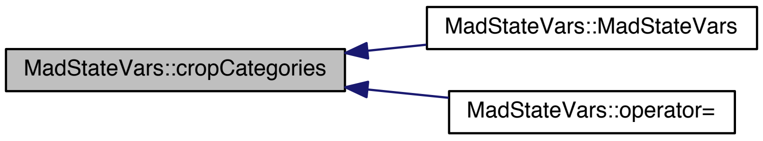
### 6.15.3 Member Function Documentation

#### 6.15.3.1 MadSVCrop MadStateVars::cropCategories ( ) const

Definition at line 64 of file madstatevars.cpp.

```
{
  return mCropCategories;
}
```

Here is the caller graph for this function:



#### 6.15.3.2 bool MadStateVars::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

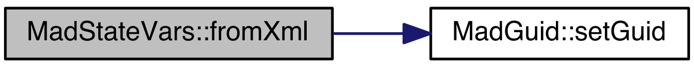
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 102 of file madstatevars.cpp.

```
{
    QDomDocument myDocument("mydocument");
    myDocument.setContent(theXml);
    QDomElement myTopElement = myDocument.firstChildElement("statevars");
    if (myTopElement.isNull())
    {
        //TODO - just make this a warning
        qDebug("the top element couldn't be found!");
        setGuid(myTopElement.attribute("guid"));
        //mName=MadUtils::xmlDecode(myTopElement.firstChildElement("name").text());
        //mDescription=MadUtils::xmlDecode(myTopElement.firstChildElement("description").text());
        //mImageFile=QString(myTopElement.firstChildElement("imageFile").text());
        return true;
    }
    else
    return false;
}
```

Here is the call graph for this function:



#### 6.15.3.3 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

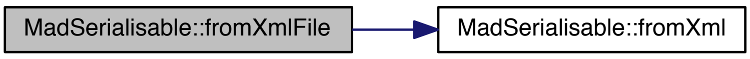
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



#### 6.15.3.4 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

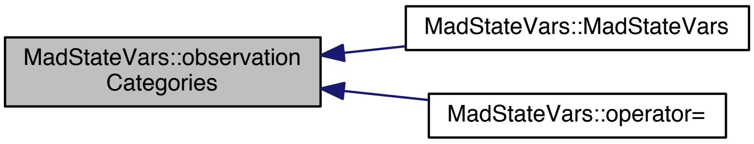
```
{
    return mGuid;
}
```

#### 6.15.3.5 MadSVObservations MadStateVars::observationCategories ( ) const

Definition at line 76 of file madstatevars.cpp.

```
{
  return mObservations;
}
```

Here is the caller graph for this function:

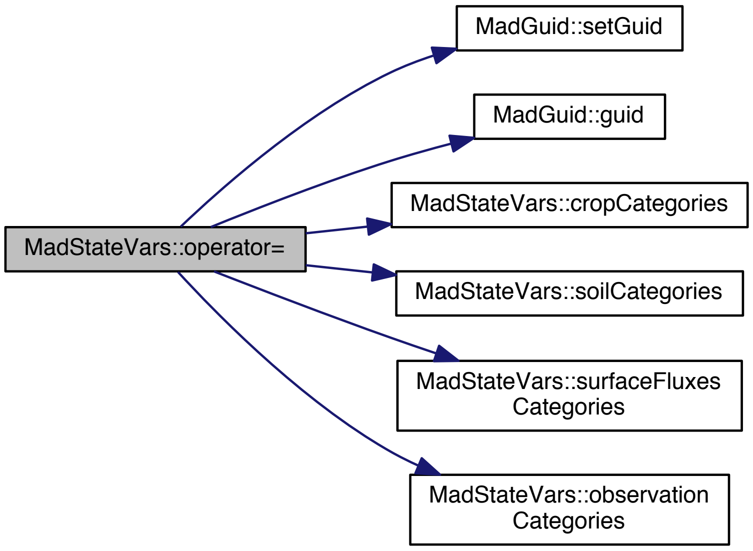

MadStateVars::observation Categories
MadStateVars::MadStateVars
MadStateVars::operator=

#### 6.15.3.6 MadStateVars & MadStateVars::operator= ( const MadStateVars & *theData* )

Definition at line 50 of file madstatevars.cpp.

```
{
  // gracefully handles self assignment
  if (this == &theData) return *this;
  setGuid(theData.guid());
  mCropCategories=theData.cropCategories();
  mSoilCategories=theData.soilCategories();
  mSurfaceFluxes=theData.surfaceFluxesCategories();
  mObservations=theData.observationCategories();
  return *this;
}
```

Here is the call graph for this function:

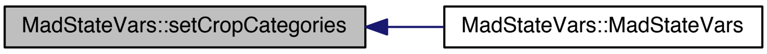

MadStateVars::operator=
MadGuid::setGuid
MadGuid::guid
MadStateVars::cropCategories
MadStateVars::soilCategories
MadStateVars::surfaceFluxes Categories
MadStateVars::observation Categories

#### 6.15.3.7 void MadStateVars::setCropCategories ( MadSVCrop *theMadSVCrop* )

Definition at line 83 of file madstatevars.cpp.

```
{
  mCropCategories = theMadSVCrop;
}
```

Here is the caller graph for this function:



#### 6.15.3.8 void MadGuid::setGuid ( QString *theGuid* = "" ) [inherited]

MadGuid::setGuid.

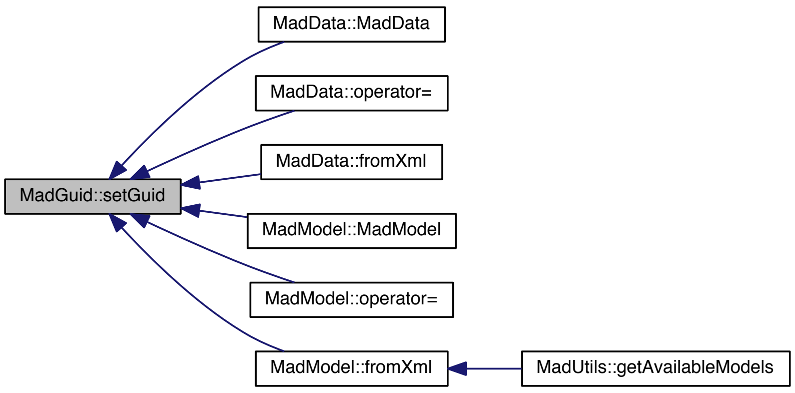
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:

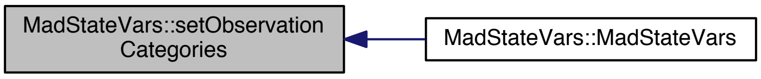


#### 6.15.3.9 void MadStateVars::setObservationCategories ( MadSVObservations *theData* )

Definition at line 97 of file madstatevars.cpp.

```
{
  mObservations = theData;
}
```

Here is the caller graph for this function:

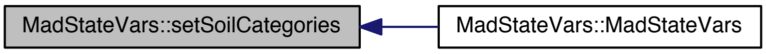

MadStateVars::setObservation Categories
MadStateVars::MadStateVars

#### 6.15.3.10 void MadStateVars::setSoilCategories ( **MadSVSoil** *theData* )

Definition at line 87 of file madstatevars.cpp.

```
{
  mSoilCategories = theData;
}
```

Here is the caller graph for this function:

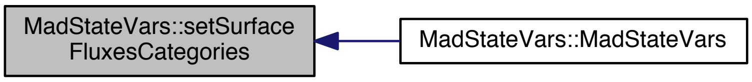


#### 6.15.3.11 void MadStateVars::setSurfaceFluxesCategories ( **MadSVSurfaceFluxes** *theData* )

Definition at line 92 of file madstatevars.cpp.

```
{
  mSurfaceFluxes = theMadSVSurfaceFluxes;
}
```

Here is the caller graph for this function:

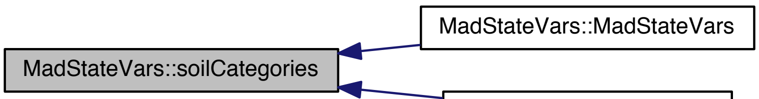

MadStateVars::setSurface FluxesCategories
MadStateVars::MadStateVars

#### 6.15.3.12 MadSVSoil MadStateVars::soilCategories ( ) const

Definition at line 68 of file madstatevars.cpp.

```
{
  return mSoilCategories;
}
```

Here is the caller graph for this function:

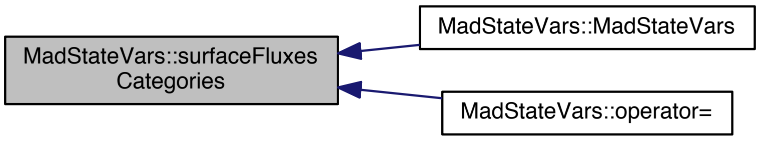


#### 6.15.3.13 MadSVSurfaceFluxes MadStateVars::surfaceFluxesCategories ( ) const

Definition at line 72 of file madstatevars.cpp.

```
{
  return mSurfaceFluxes;
}
```

Here is the caller graph for this function:


MadStateVars::surfaceFluxes Categories
MadStateVars::MadStateVars
MadStateVars::operator=

#### 6.15.3.14 QString MadStateVars::toHtml ( )

Return a html text representation of this layer

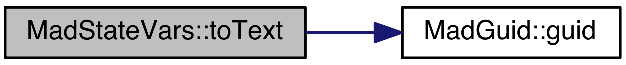
#### 6.15.3.15 QString MadStateVars::toText ( )

Return a plain text representation of this layer

Definition at line 147 of file madstatevars.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  //myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.15.3.16 QString MadStateVars::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

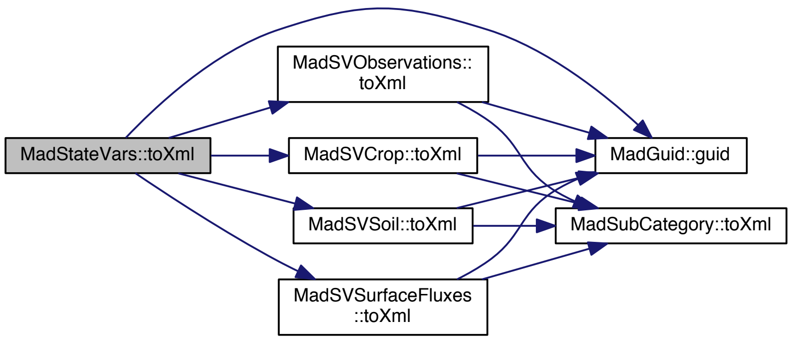
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

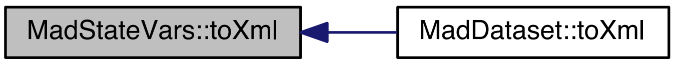
Definition at line 121 of file madstatevars.cpp.

```
{
  QString myString;
  myString+=QString("  <statevars guid=\"" + guid() + "\">\n");

//  myString+=QString("     <crop>\n");
  myString+=mCropCategories.toXml();
//  myString+=QString("     </crop>\n");

//  myString+=QString("     <soil>\n");
  myString+=mSoilCategories.toXml();
//  myString+=QString("     </soil>\n");

//  myString+=QString("     <surfacefluxes>\n");
  myString+=mSurfaceFluxes.toXml();
//  myString+=QString("     </surfacefluxes>\n");

//  myString+=QString("     <observations>\n");
  myString+=mObservations.toXml();
//  myString+=QString("     </observations>\n");

  myString+=QString("  </statevars>\n");
  return myString;

}
```

Here is the call graph for this function:



Here is the caller graph for this function:



### 6.15.3.17 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

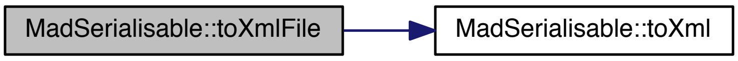
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



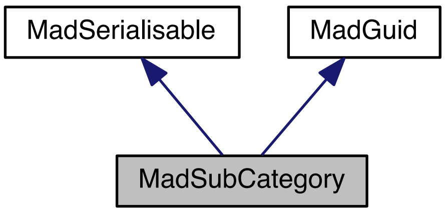
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madstatevars.-h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madstatevars.-cpp

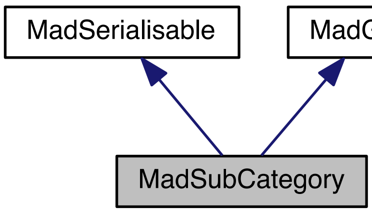
## 6.16 MadSubCategory Class Reference

`#include <madsubcategory.h>`

Inheritance diagram for MadSubCategory:



Collaboration diagram for MadSubCategory:



## Public Member Functions

- MadSubCategory ()
- MadSubCategory (const MadSubCategory &theSubCategory)
- MadSubCategory & operator= (const MadSubCategory &theData)
- bool minData () const
- float depth () const
- int observations () const
- float weightPoints () const
- int replicates () const
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- void setMinData (bool theBool)
- void setDepth (float theValue)
- void setObservations (int theValue)
- void setWeightPoints (float theValue)
- void setReplicates (int theValue)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.16.1 Detailed Description

Definition at line 32 of file madsubcategory.h.

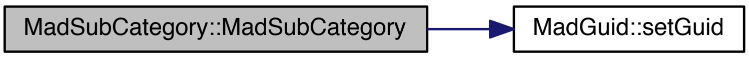
### 6.16.2 Constructor & Destructor Documentation

#### 6.16.2.1 MadSubCategory::MadSubCategory ( )

Definition at line 33 of file madsubcategory.cpp.

```
                                   : MadSerialisable(), MadGuid()
{
  setGuid();
  mMinData= 0;
  mDepth = 0.0;
  mObservations = 0;
  mWeightPoints = 0.0;
  mReplicates = 0;
}
```

Here is the call graph for this function:

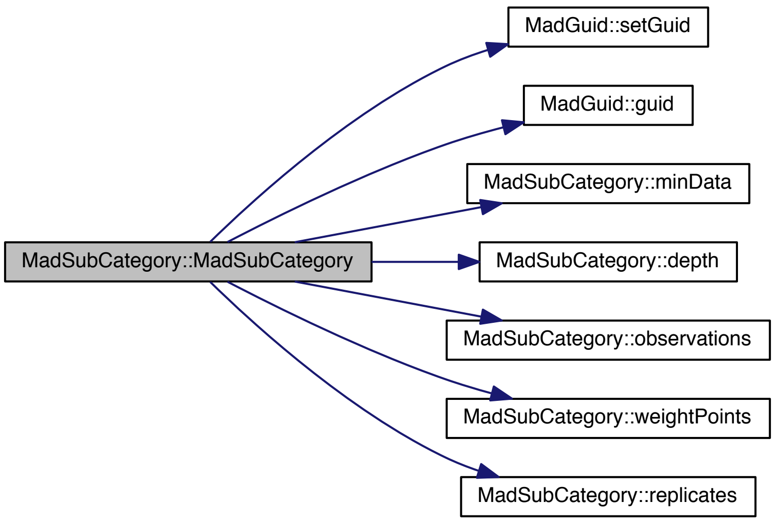


#### 6.16.2.2 MadSubCategory::MadSubCategory ( const MadSubCategory & *theSubCategory* )

Definition at line 44 of file madsubcategory.cpp.

```
{
  setGuid(theSubCategory.guid());
  mMinData = theSubCategory.minData();
  mDepth = theSubCategory.depth();
  mObservations = theSubCategory.observations();
  mWeightPoints = theSubCategory.weightPoints();
  mReplicates = theSubCategory.replicates();
}
```

Here is the call graph for this function:


MadSubCategory::MadSubCategory
MadGuid::setGuid
MadGuid::guid
MadSubCategory::minData
MadSubCategory::depth
MadSubCategory::observations
MadSubCategory::weightPoints
MadSubCategory::replicates

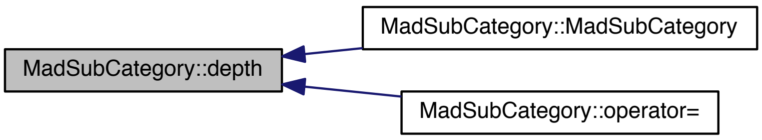
### 6.16.3 Member Function Documentation

#### 6.16.3.1 float MadSubCategory::depth ( ) const

Definition at line 74 of file madsubcategory.cpp.

```
{
  return mDepth;
}
```

Here is the caller graph for this function:



#### 6.16.3.2 bool MadSubCategory::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

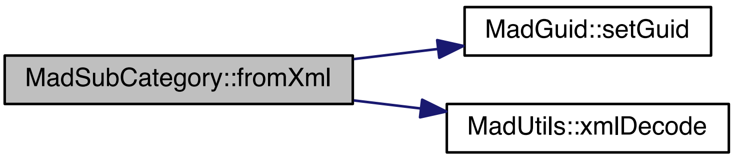
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 120 of file madsubcategory.cpp.

```
{
  QDomDocument myDocument("mydocument");
  myDocument.setContent(theXml);
  QDomElement myTopElement = myDocument.firstChildElement("details");
  if (myTopElement.isNull())
  {
    // TODO - just make this a warning
    qDebug("the top element couldn't be found!");
    setGuid(myTopElement.attribute("guid"));

    // the line below works and does the same as the line below it.
    // (QString(myTopElement.firstChildElement("mindata").text() ))=="0" ? mMinData=false : mMinData=true;
    mMinData = QString(myTopElement.firstChildElement("mindata").text()).toInt();

    mDepth = MadUtils::xmlDecode(myTopElement.firstChildElement("depth").text()).toFloat();
    mObservations = MadUtils::xmlDecode(myTopElement.firstChildElement("observations").text()).toInt();
    mWeightPoints = MadUtils::xmlDecode(myTopElement.firstChildElement("weightpoints").text()).toFloat();
    mReplicates = MadUtils::xmlDecode(myTopElement.firstChildElement("replicates").text()).toInt();

    return true;
  }
  else
    return false;
}
```

Here is the call graph for this function:



#### 6.16.3.3 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

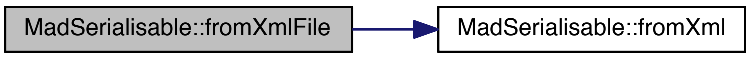
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



#### 6.16.3.4 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

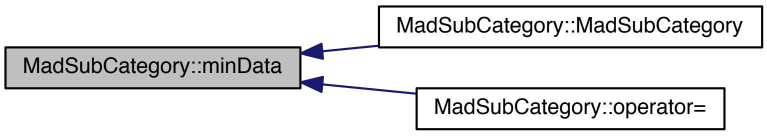
```
{
    return mGuid;
}
```

#### 6.16.3.5 bool MadSubCategory::minData ( ) const

Definition at line 69 of file madsubcategory.cpp.

```
{
  return mMinData;
}
```

Here is the caller graph for this function:

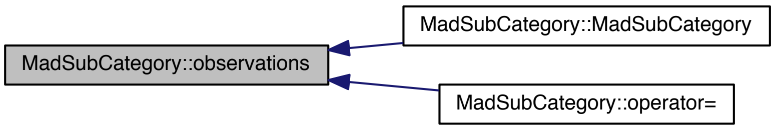


#### 6.16.3.6 int MadSubCategory::observations ( ) const

Definition at line 79 of file madsubcategory.cpp.

```
{
  return mObservations;
}
```

Here is the caller graph for this function:

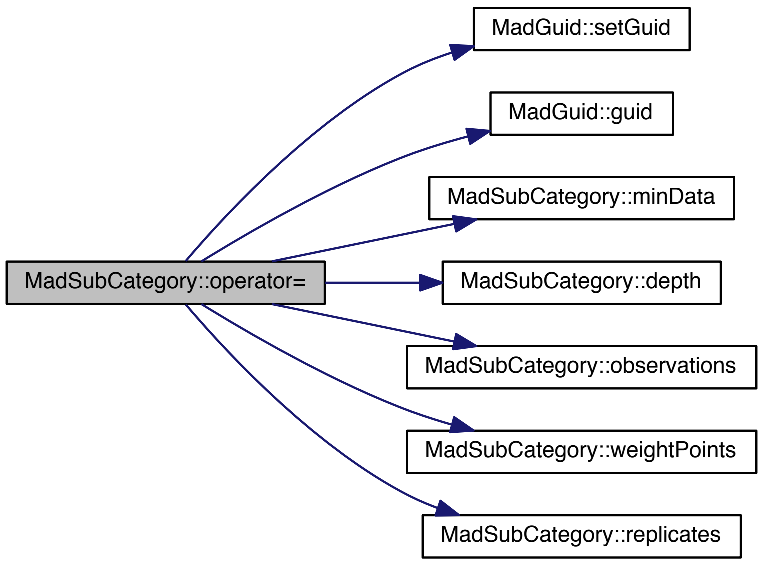

MadSubCategory::observations
MadSubCategory::MadSubCategory
MadSubCategory::operator=

#### 6.16.3.7 MadSubCategory & MadSubCategory::operator= ( const MadSubCategory & *theData* )

Definition at line 54 of file madsubcategory.cpp.

```
{
  // gracefully handles self assignment
  if (this == &theSubCategory) return *this;
  setGuid(theSubCategory.guid());
  mMinData = theSubCategory.minData();
  mDepth = theSubCategory.depth();
  mObservations = theSubCategory.observations();
  mWeightPoints = theSubCategory.weightPoints();
  mReplicates = theSubCategory.replicates();
  return *this;
}
```

Here is the call graph for this function:

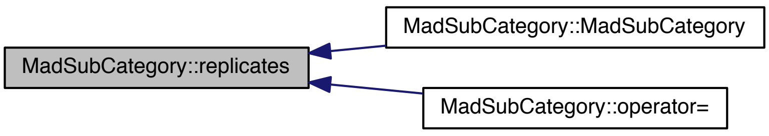

MadGuid::setGuid
MadGuid::guid
MadSubCategory::minData
MadSubCategory::operator=
MadSubCategory::depth
MadSubCategory::observations
MadSubCategory::weightPoints
MadSubCategory::replicates

#### 6.16.3.8 int MadSubCategory::replicates ( ) const

Definition at line 89 of file madsubcategory.cpp.

```
{
  return mReplicates;
}
```

Here is the caller graph for this function:

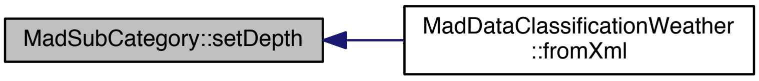

MadSubCategory::replicates
MadSubCategory::MadSubCategory
MadSubCategory::operator=

#### 6.16.3.9 void MadSubCategory::setDepth ( float *theValue* )

Definition at line 100 of file madsubcategory.cpp.

```
{
  mDepth = theValue;
}
```

Here is the caller graph for this function:



### 6.16.3.10 void MadGuid::setGuid ( QString *theGuid* = " " ) [inherited]

MadGuid::setGuid.

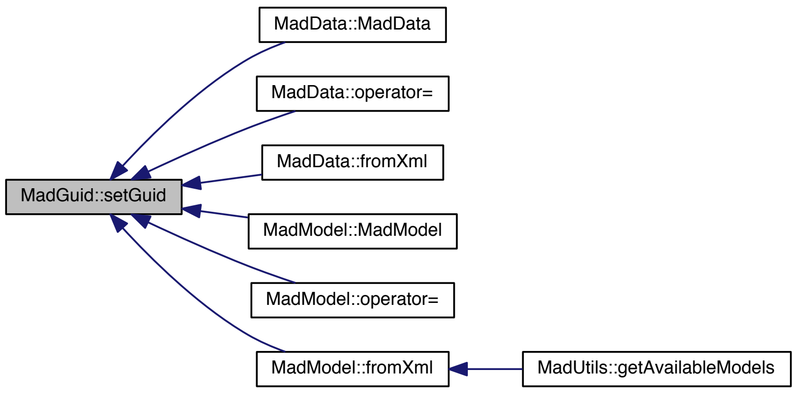
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:



### 6.16.3.11 void MadSubCategory::setMinData ( bool *theBool* )

Definition at line 95 of file madsubcategory.cpp.

```
{
  mMinData = theBool;
}
```

#### 6.16.3.12 void MadSubCategory::setObservations ( int *theValue* )

Definition at line 105 of file madsubcategory.cpp.

```
{
  mObservations = theValue;
}
```

#### 6.16.3.13 void MadSubCategory::setReplicates ( int *theValue* )

Definition at line 115 of file madsubcategory.cpp.

```
{
  mReplicates = theValue;
}
```

#### 6.16.3.14 void MadSubCategory::setWeightPoints ( float *theValue* )

Definition at line 110 of file madsubcategory.cpp.

```
{
  mWeightPoints = theValue;
}
```

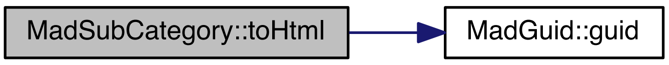
#### 6.16.3.15 QString MadSubCategory::toHtml ( )

Return a html text representation of this layer

Definition at line 175 of file madsubcategory.cpp.

```
{
  QString myString;
  myString+="<h3>Details for values in the form:</h3>";
  myString+="<table>";
  //myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";
  myString+="<p>GUID:" + guid() + "</p>";


  QString myMinData = (mMinData==0) ? "false" : "true";

  myString+="<tr><td><b>min. Data: </b></td><td>" + myMinData + "</td></tr>";
  myString+="<tr><td><b>Depth: </b></td><td>" + QString::number(mDepth) + "</td></tr>";
  myString+="<tr><td><b>Observations: </b></td><td>" + QString::number(mObservations) + "</td></tr>";
  myString+="<tr><td><b>WeightPoints: </b></td><td>" + QString::number(mWeightPoints) + "</td></tr>";
  myString+="<tr><td><b>Replicates: </b></td><td>" + QString::number(mReplicates) + "</td></tr>";

  myString+="</table>";
  return myString;
}
```

Here is the call graph for this function:



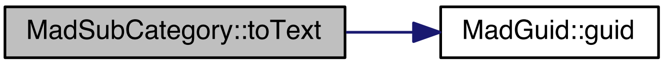
### 6.16.3.16 QString MadSubCategory::toText ( )

Return a plain text representation of this layer

Definition at line 160 of file madsubcategory.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  //myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  //myString+=QString("<dataset guid=\"" + guid() + "\">\n");
  myString+=QString("minData=>" + QString::number(mMinData) + "\n");
  myString+=QString("depth=>" + QString::number(mDepth) + "\n");
  myString+=QString("observations=>" + QString::number(mObservations) + "<\n");
  myString+=QString("weightPoints=>" + QString::number(mWeightPoints) + "<\n");
  myString+=QString("replicates=>" + QString::number(mReplicates) + "<\n");
  return myString;
}
```

Here is the call graph for this function:



### 6.16.3.17 QString MadSubCategory::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

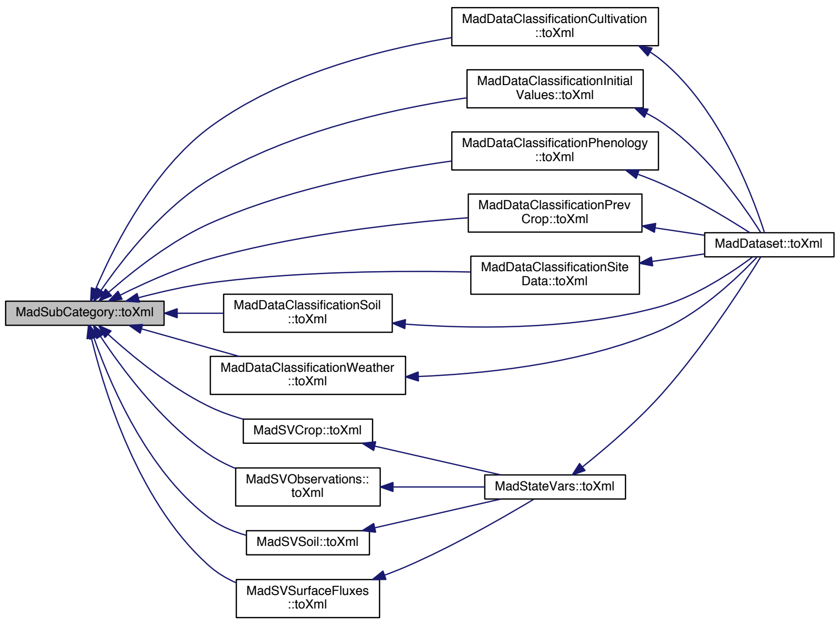
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 146 of file madsubcategory.cpp.

```
{
  QString myString;
  myString+=QString("       <details>\n");
  myString+=QString("        <mindata>" + QString::number(mMinData) + "</mindata>\n");
  myString+=QString("        <depth>" + QString::number(mDepth) + "</depth>\n");
  myString+=QString("        <observations>" + QString::number(mObservations) + "</observations>\n");
  myString+=QString("        <weightpoints>" + QString::number(mWeightPoints) + "</weightpoints>\n");
  myString+=QString("        <replicates>" + QString::number(mReplicates) + "</replicates>\n");
  myString+=QString("       </details>\n");

  return myString;
}
```

Here is the caller graph for this function:


MadDataClassificationCultivation ::toXml
MadDataClassificationInitial Values::toXml
MadDataClassificationPhenology ::toXml
MadDataClassificationPrev Crop::toXml
MadDataClassificationSite Data::toXml
MadDataset::toXml
MadSubCategory::toXml
MadDataClassificationSoil ::toXml
MadDataClassificationWeather ::toXml
MadSVCrop::toXml
MadSVObservations:: toXml
MadStateVars::toXml
MadSVSoil::toXml
MadSVSurfaceFluxes ::toXml

### 6.16.3.18 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

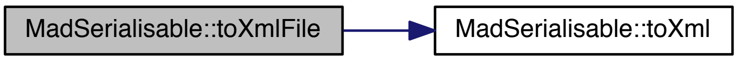
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:

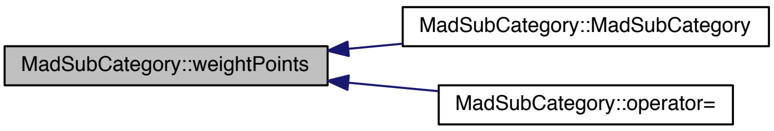


#### 6.16.3.19 float MadSubCategory::weightPoints ( ) const

Definition at line 84 of file madsubcategory.cpp.

```
{
  return mWeightPoints;
}
```

Here is the caller graph for this function:


MadSubCategory::weightPoints
MadSubCategory::MadSubCategory
MadSubCategory::operator=

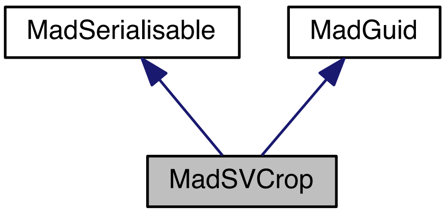
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/madsubcategory.h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/madsubcategory.cpp

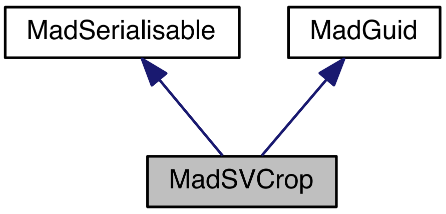
## 6.17 MadSVCrop Class Reference

`#include <madsvcrop.h>`

Inheritance diagram for MadSVCrop:



Collaboration diagram for MadSVCrop:



## Public Member Functions

- MadSVCrop ()
- MadSVCrop (const MadSVCrop &theData)
- MadSVCrop & operator= (const MadSVCrop &theData)
- MadSubCategory agrBiomass () const

  *agrBiomass*
- MadSubCategory weightOrgans () const

  *weightOrgans*
- MadSubCategory rootBiomass () const

  *rootBiomass*
- MadSubCategory nlnAGrBiomass () const

  *nlnAGrBiomass*
- MadSubCategory nlnOrgans () const

  *nlnOrgans*
- MadSubCategory lai () const

  *lai*
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)

- void setAgrBiomass (MadSubCategory theData)

  *setAgrBiomass*
- void setWeightOrgans (MadSubCategory theData)

  *setWeightOrgans*
- void setRootBiomass (MadSubCategory theData)

  *setRootBiomass*
- void setNInAGrBiomass (MadSubCategory theData)

  *setNInAGrBiomass*
- void setNInOrgans (MadSubCategory theData)

  *setNInOrgans*
- void setLai (MadSubCategory theData)

  *setLai*
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.17.1 Detailed Description

Definition at line 35 of file madsvcrop.h.

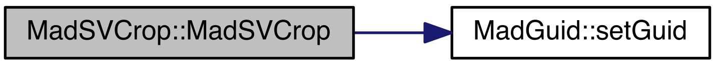
### 6.17.2 Constructor & Destructor Documentation

#### 6.17.2.1 MadSVCrop::MadSVCrop ( )

Definition at line 33 of file madsvcrop.cpp.

```
                      : MadSerialisable(), MadGuid()
{
  setGuid();
}
```

Here is the call graph for this function:

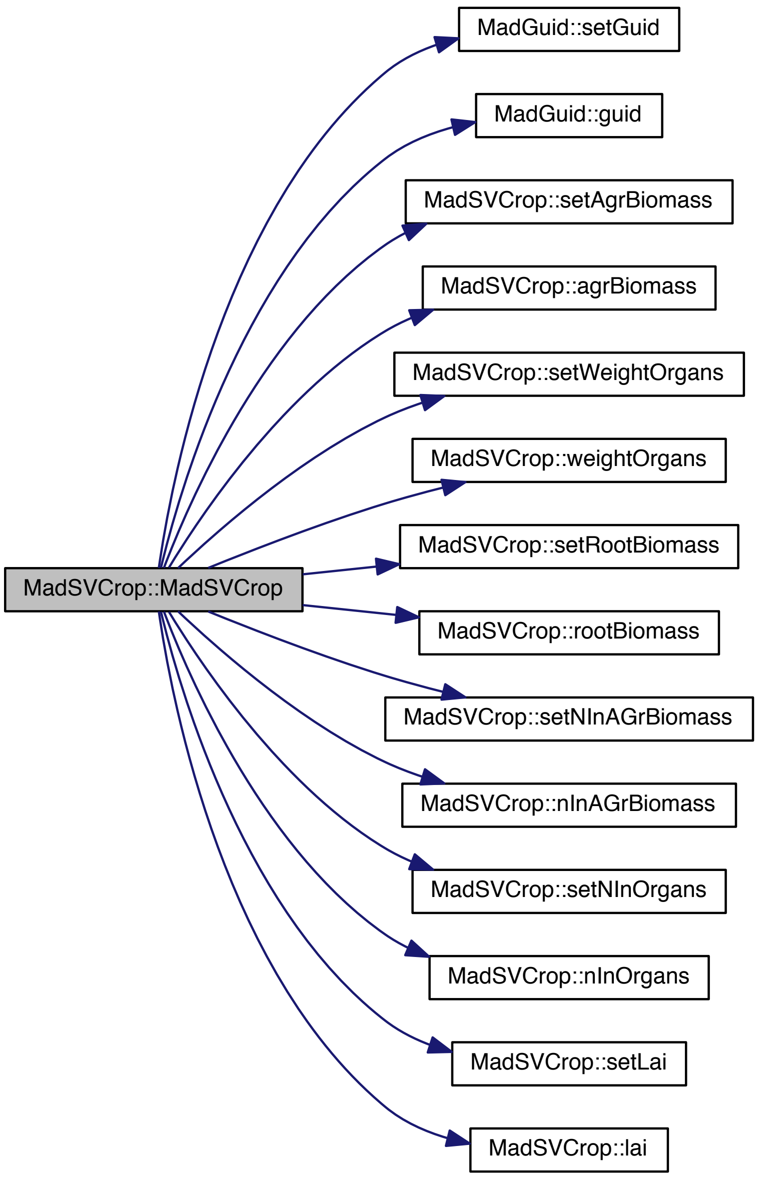


#### 6.17.2.2 MadSVCrop::MadSVCrop ( const **MadSVCrop** & *theData* )

Definition at line 38 of file madsvcrop.cpp.

```
{
  setGuid(theData.guid());
  setAgrBiomass(theData.agrBiomass());
  setWeightOrgans(theData.weightOrgans());
  setRootBiomass(theData.rootBiomass());
  setNInAGrBiomass(theData.nInAGrBiomass());
  setNInOrgans(theData.nInOrgans());
  setLai(theData.lai());
}
```

Here is the call graph for this function:



## 6.17.3 Member Function Documentation

### 6.17.3.1 MadSubCategory MadSVCrop::agrBiomass ( ) const

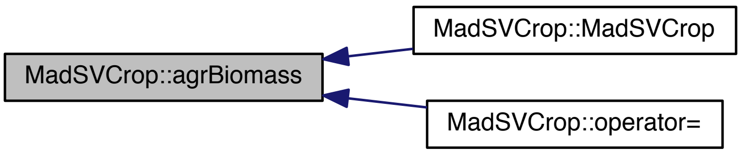
agrBiomass

**Returns**

Definition at line 64 of file madsvcrop.cpp.

```
{
  return mAgrBiomass;
}
```

Here is the caller graph for this function:



### 6.17.3.2 bool MadSVCrop::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

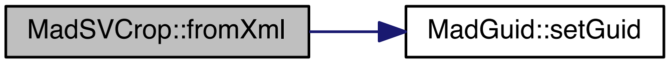
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 125 of file madsvcrop.cpp.

```
{
    QDomDocument myDocument("mydocument");
    myDocument.setContent(theXml);
    QDomElement myTopElement = myDocument.firstChildElement("svcrop");
    if (myTopElement.isNull())
    {
        //TODO - just make this a warning
        qDebug("the top element couldn't be found!");
        setGuid(myTopElement.attribute("guid"));
        //QDomElement myCategory;
        //QDomElement myDetails;

        //myCategory=QString(myTopElement.firstChildElement("agrbiomass").text());
        //myDetails=QString(myCategory.firstChildElement("details").text());
        //mWeightOrgans.depth()=QString(myDetails.firstChildElement("weightorgans").text()).toFloat();

        //MadSubCategory mySVCropDetails;
        //mySVCropDetails = QString(myTopElement.firstChildElement("details").text.());

        //mName=MadUtils::xmlDecode(myTopElement.firstChildElement("name").text());
        //mDescription=MadUtils::xmlDecode(myTopElement.firstChildElement("description").text());
        //mImageFile=QString(myTopElement.firstChildElement("imageFile").text());
        return true;
    }
    else
    return false;
}
```

Here is the call graph for this function:



### 6.17.3.3 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

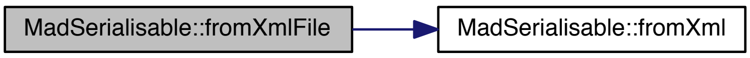
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



#### 6.17.3.4 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

```
{
    return mGuid;
}
```

#### 6.17.3.5 MadSubCategory MadSVCrop::lai ( ) const

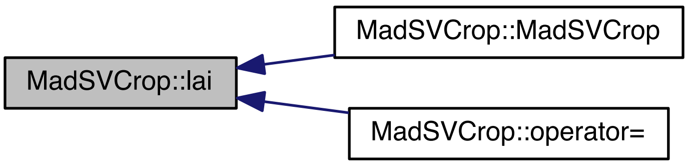
lai

**Returns**

Definition at line 88 of file madsvcrop.cpp.

```
{
  return mLai;
}
```

Here is the caller graph for this function:



#### 6.17.3.6 MadSubCategory MadSVCrop::nInAGrBiomass ( ) const

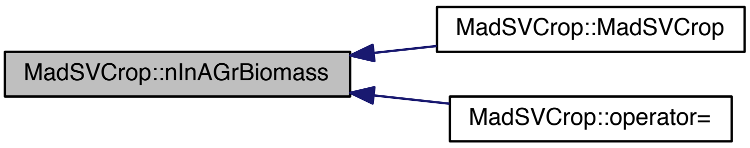
nInAGrBiomass

**Returns**

Definition at line 78 of file madsvcrop.cpp.

```
{
  return mNInAGrBiomass;
}
```

Here is the caller graph for this function:



#### 6.17.3.7 MadSubCategory MadSVCrop::nInOrgans ( ) const

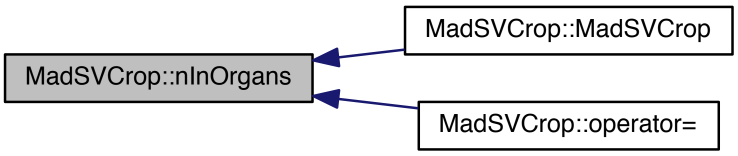
nInOrgans

**Returns**

Definition at line 83 of file madsvcrop.cpp.

```
{
  return mNInOrgans;
}
```

Here is the caller graph for this function:

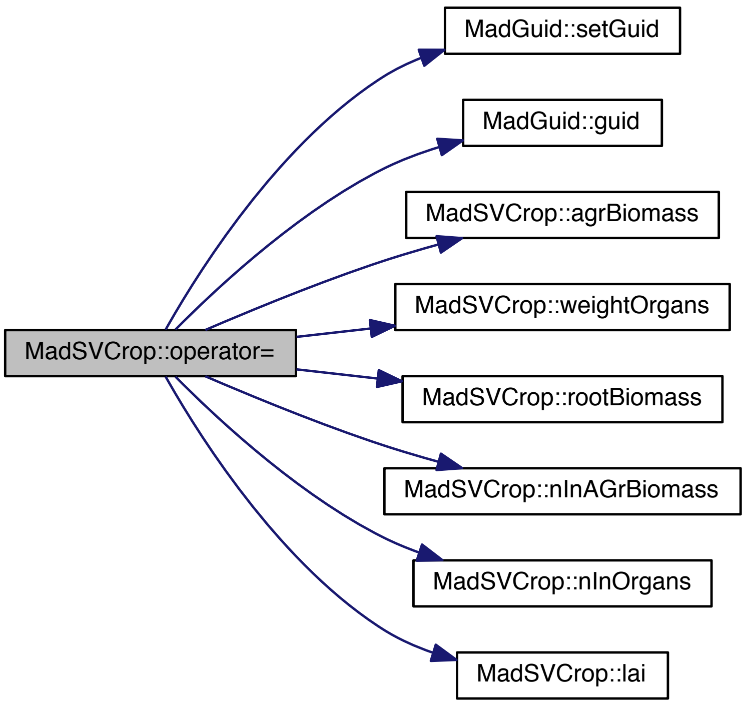


#### 6.17.3.8 MadSVCrop & MadSVCrop::operator= ( const MadSVCrop & *theData* )

Definition at line 49 of file madsvcrop.cpp.

```
{
  // gracefully handles self assignment
  if (this == &theData) return *this;
  setGuid(theData.guid());
  mAgrBiomass=theData.agrBiomass();
  mWeightOrgans=theData.weightOrgans();
  mRootBiomass=theData.rootBiomass();
  mNInAGrBiomass=theData.nInAGrBiomass();
  mNInOrgans=theData.nInOrgans();
  mLai=theData.lai();
  return *this;
}
```

Here is the call graph for this function:



#### 6.17.3.9 MadSubCategory MadSVCrop::rootBiomass ( ) const

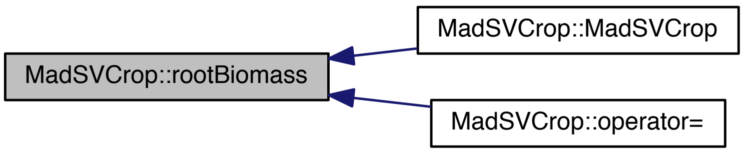
rootBiomass

**Returns**

Definition at line 73 of file madsvcrop.cpp.

```
{
  return mRootBiomass;
}
```

Here is the caller graph for this function:



#### 6.17.3.10 void MadSVCrop::setAgrBiomass ( MadSubCategory *theData* )

setAgrBiomass

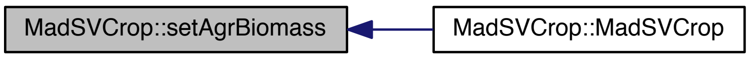
**Parameters**

| | |
|---|---|
| *theData* | |

Definition at line 94 of file madsvcrop.cpp.

```
{
  mAgrBiomass = theData;
}
```

Here is the caller graph for this function:



#### 6.17.3.11 void MadGuid::setGuid ( QString *theGuid* = " " ) [inherited]

MadGuid::setGuid.

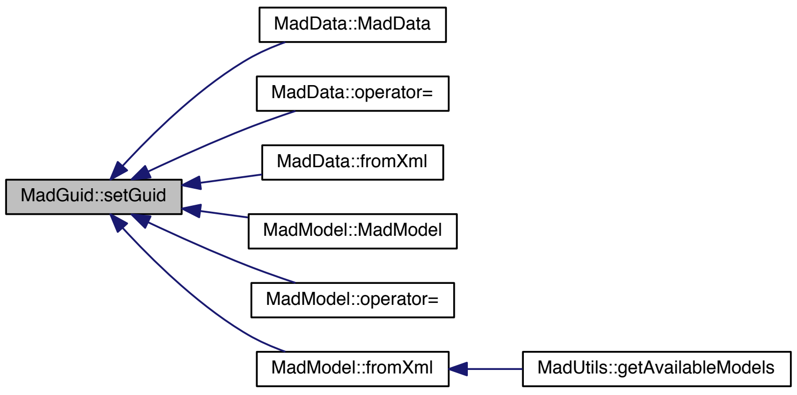
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:



#### 6.17.3.12 void MadSVCrop::setLai ( **MadSubCategory** *theData* )

setLai

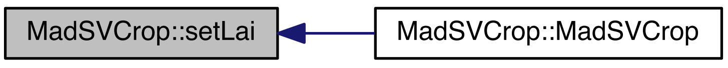
**Parameters**

| | |
|---|---|
| *theData* | |

Definition at line 119 of file madsvcrop.cpp.

```
{
  mLai = theData;
}
```

Here is the caller graph for this function:



#### 6.17.3.13 void MadSVCrop::setNInAGrBiomass ( **MadSubCategory** *theData* )

setNInAGrBiomass

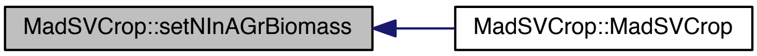
**Parameters**

| | |
|---|---|
| *theData* | |

Definition at line 109 of file madsvcrop.cpp.

```
{
  mNInAGrBiomass = theData;
}
```

Here is the caller graph for this function:



### 6.17.3.14 void MadSVCrop::setNInOrgans ( **MadSubCategory** *theData* )

setNInOrgans

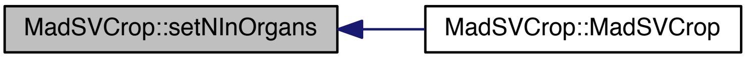
**Parameters**

| | |
|---|---|
| *theData* | |

Definition at line 114 of file madsvcrop.cpp.

```
{
  mNInOrgans = theData;
}
```

Here is the caller graph for this function:



### 6.17.3.15 void MadSVCrop::setRootBiomass ( **MadSubCategory** *theData* )

setRootBiomass

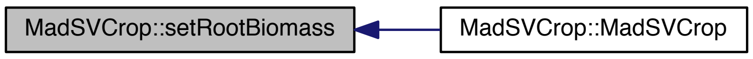
**Parameters**

| | |
|---|---|
| *theData* | |

Definition at line 104 of file madsvcrop.cpp.

```
{
  mRootBiomass = theData;
}
```

Here is the caller graph for this function:



### 6.17.3.16 void MadSVCrop::setWeightOrgans ( **MadSubCategory** *theData* )

setWeightOrgans

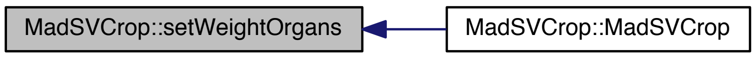
**Parameters**

| | |
|---|---|
| *theData* | |

Definition at line 99 of file madsvcrop.cpp.

```
{
  mWeightOrgans = theData;
}
```

Here is the caller graph for this function:



### 6.17.3.17 QString MadSVCrop::toHtml ( )

Return a html text representation of this layer

Definition at line 198 of file madsvcrop.cpp.

```
{
  QString myString;
  //myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  //myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";

  //
  // the following shows example of how to do a couple of things
  //

  //myString+="<tr><td><b>Cals/Kg: </b></td><td>" + QString::number(mCropCalories) + "</td></tr>";
  //QString myCropFodderEnergyType = (mCropFodderEnergyType==0) ? "KCalories" : "TDN";
  //QString myUnits = (mAreaUnits==0) ? "Dunum" : "Hectare";
  //myString+="<tr><td><b>Fodder (kg/" + myUnits + "): </b></td><td>" +
   QString::number(mCropFodderProduction) + "</td></tr>";
  //myString+="<tr><td><b>Fodder Value/Kg: </b></td><td>" + QString::number(mCropFodderValue) +
   "</td></tr>";
  //myString+="<tr><td><b>FodderEnergyType: </b></td><td>" + myCropFodderEnergyType + "</td></tr>";
  //myString+="<tr><td><b>AreaUnits: </b></td><td>" + myUnits + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

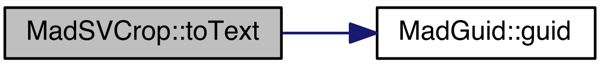
#### 6.17.3.18 QString MadSVCrop::toText ( )

Return a plain text representation of this layer

Definition at line 189 of file madsvcrop.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  //myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.17.3.19 QString MadSVCrop::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

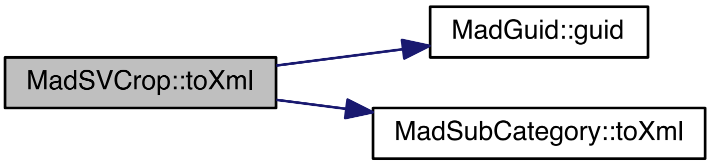
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

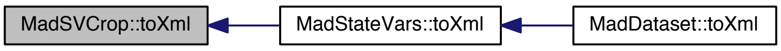
Definition at line 154 of file madsvcrop.cpp.

```
{
  QString myString;
  myString+=QString("    <svcrop guid=\"" + guid() + "\">\n");

  myString+=QString("     <agrbiomass>\n");
  myString+=mAgrBiomass.toXml();
  myString+=QString("     </agrbiomass>\n");

  myString+=QString("     <weightorgans>\n");
  myString+=mWeightOrgans.toXml();
  myString+=QString("     </weightorgans>\n");

  myString+=QString("     <rootbiomass>\n");
  myString+=mRootBiomass.toXml();
  myString+=QString("     </rootbiomass>\n");

  myString+=QString("     <ninagrbiomass>\n");
  myString+=mNInAGrBiomass.toXml();
  myString+=QString("     </ninagrbiomass>\n");

  myString+=QString("     <ninorgans>\n");
  myString+=mNInOrgans.toXml();
  myString+=QString("     </ninorgans>\n");

  myString+=QString("      <lai>\n");
  myString+=mLai.toXml();
  myString+=QString("      </lai>\n");

  myString+=QString("    </svcrop>\n");
  return myString;


}
```

Here is the call graph for this function:



Here is the caller graph for this function:



#### 6.17.3.20 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

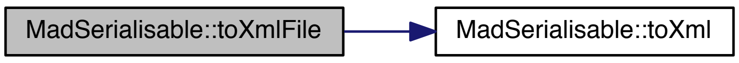
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



#### 6.17.3.21 **MadSubCategory** MadSVCrop::weightOrgans ( ) const

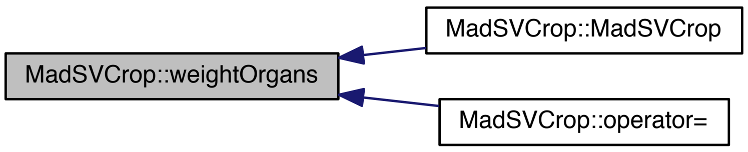
weightOrgans

**Returns**

Definition at line 68 of file madsvcrop.cpp.

```
{
  return mWeightOrgans;
}
```

Here is the caller graph for this function:



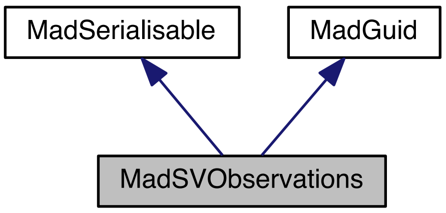
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madsvcrop.h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madsvcrop.-cpp

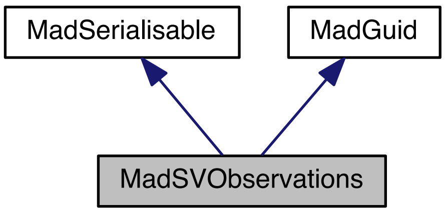
## 6.18 MadSVObservations Class Reference

```
#include <madsvobservations.h>
```

Inheritance diagram for MadSVObservations:



Collaboration diagram for MadSVObservations:



### Public Member Functions

- MadSVObservations ()
- MadSVObservations (const MadSVObservations &theData)
- MadSVObservations & operator= (const MadSVObservations &theData)
- MadSubCategory lodging () const
- MadSubCategory pestsOrDiseases () const
- MadSubCategory damages () const
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- void setLodging (MadSubCategory theData)
- void setPestsOrDiseases (MadSubCategory theData)
- void setDamages (MadSubCategory theData)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*

- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*

- QString guid () const

  *MadGuid::guid.*

- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.18.1 Detailed Description

Definition at line 35 of file madsvobservations.h.

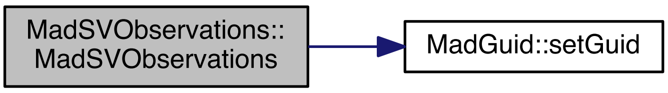
### 6.18.2 Constructor & Destructor Documentation

#### 6.18.2.1 MadSVObservations::MadSVObservations ( )

Definition at line 33 of file madsvobservations.cpp.

```
                                        : MadSerialisable(), MadGuid()
{
  setGuid();
}
```

Here is the call graph for this function:

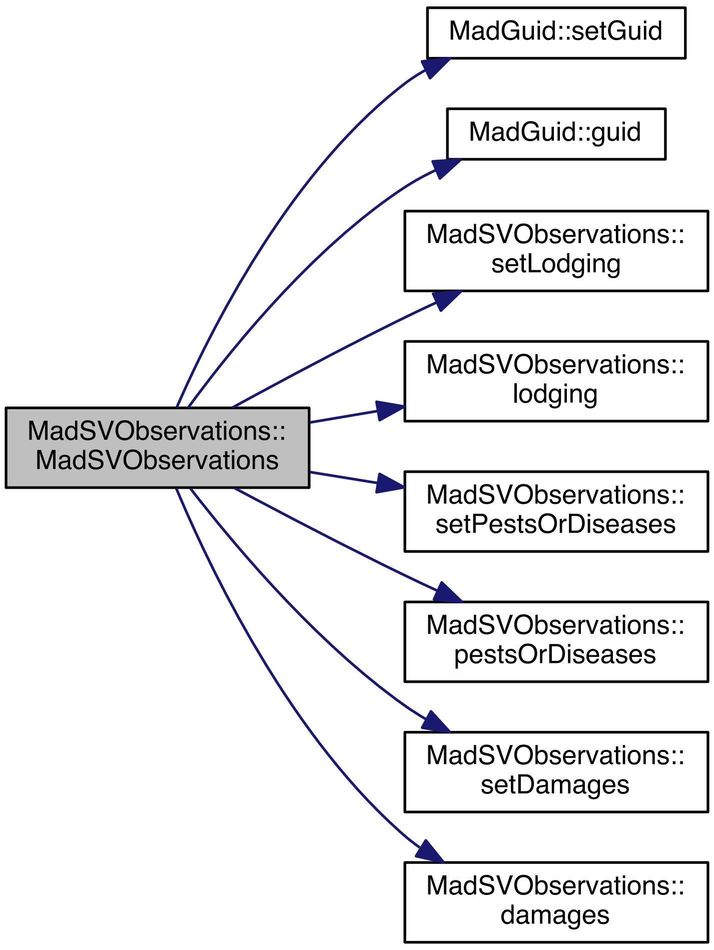


#### 6.18.2.2 MadSVObservations::MadSVObservations ( const MadSVObservations & *theData* )

Definition at line 38 of file madsvobservations.cpp.

```
{
  setGuid(theData.guid());
  setLodging(theData.lodging());
  setPestsOrDiseases(theData.pestsOrDiseases());
  setDamages(theData.damages());
}
```

Here is the call graph for this function:


MadGuid::setGuid
MadGuid::guid
MadSVObservations::setLodging
MadSVObservations::lodging
MadSVObservations::MadSVObservations
MadSVObservations::setPestsOrDiseases
MadSVObservations::pestsOrDiseases
MadSVObservations::setDamages
MadSVObservations::damages

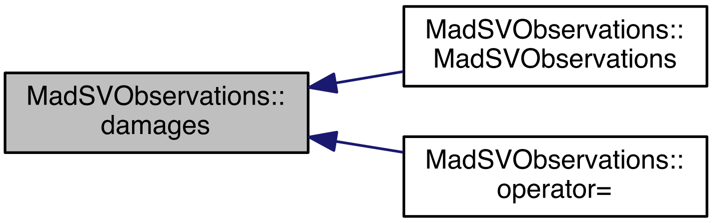
### 6.18.3 Member Function Documentation

#### 6.18.3.1 MadSubCategory MadSVObservations::damages ( ) const

Definition at line 66 of file madsvobservations.cpp.

```
{
  return mDamages;
}
```

Here is the caller graph for this function:



#### 6.18.3.2 bool MadSVObservations::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

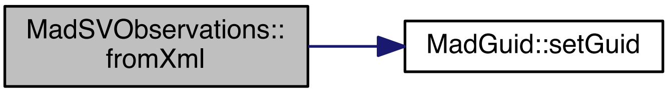
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 87 of file madsvobservations.cpp.

```
{
    QDomDocument myDocument("mydocument");
    myDocument.setContent(theXml);
    QDomElement myTopElement = myDocument.firstChildElement("svobservations");
    if (myTopElement.isNull())
    {
        //TODO - just make this a warning
        qDebug("the top element couldn't be found!");
        setGuid(myTopElement.attribute("guid"));
        //mName=MadUtils::xmlDecode(myTopElement.firstChildElement("name").text());
        //mDescription=MadUtils::xmlDecode(myTopElement.firstChildElement("description").text());
        //mImageFile=QString(myTopElement.firstChildElement("imageFile").text());
         return true;
     }
     else
     return false;
 }
```

Here is the call graph for this function:



#### 6.18.3.3 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

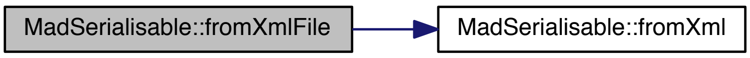
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



#### 6.18.3.4 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

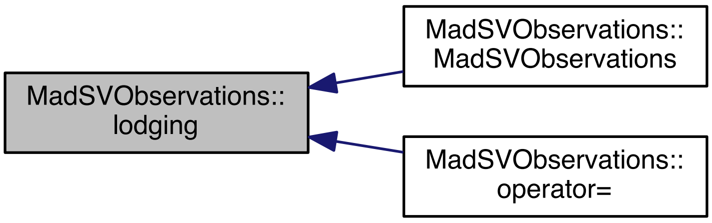
```
{
    return mGuid;
}
```

#### 6.18.3.5 MadSubCategory MadSVObservations::lodging ( ) const

Definition at line 58 of file madsvobservations.cpp.

```
{
  return mLodging;
}
```

Here is the caller graph for this function:

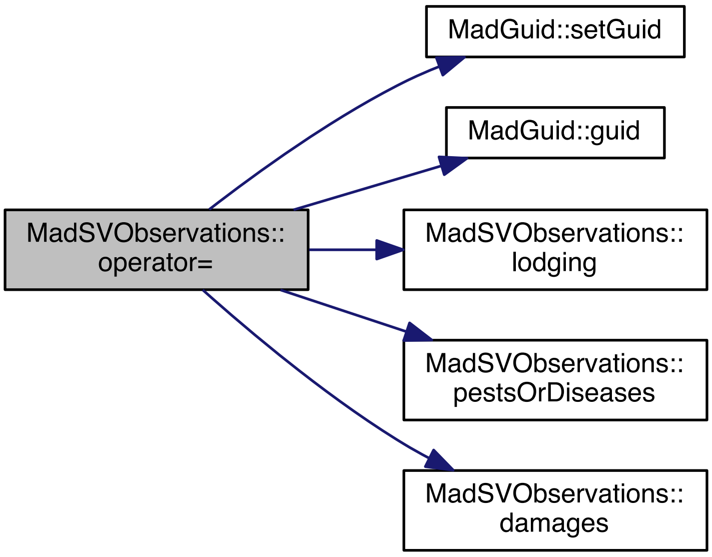


#### 6.18.3.6 MadSVObservations & MadSVObservations::operator= ( const MadSVObservations & *theData* )

Definition at line 46 of file madsvobservations.cpp.

```
{
  // gracefully handles self assignment
  if (this == &theData) return *this;
  setGuid(theData.guid());
  mLodging=theData.lodging();
  mPestsOrDiseases=theData.pestsOrDiseases();
  mDamages=theData.damages();
  return *this;
}
```

Here is the call graph for this function:

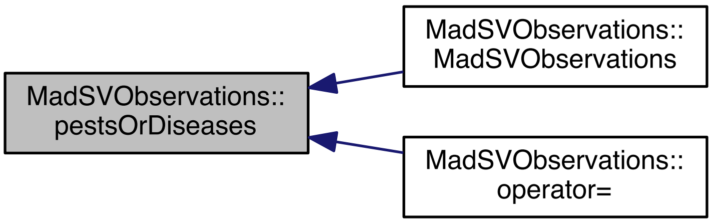

MadGuid::setGuid
MadGuid::guid
MadSVObservations:: operator=
MadSVObservations:: lodging
MadSVObservations:: pestsOrDiseases
MadSVObservations:: damages

#### 6.18.3.7 MadSubCategory MadSVObservations::pestsOrDiseases ( ) const

Definition at line 62 of file madsvobservations.cpp.

```
{
  return mPestsOrDiseases;
}
```

Here is the caller graph for this function:

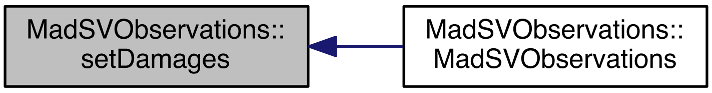

MadSVObservations:: pestsOrDiseases
MadSVObservations:: MadSVObservations
MadSVObservations:: operator=

#### 6.18.3.8 void MadSVObservations::setDamages ( MadSubCategory *theData* )

Definition at line 82 of file madsvobservations.cpp.

```
{
  mDamages = theData;
}
```

Here is the caller graph for this function:


MadSVObservations::setDamages
MadSVObservations::MadSVObservations

#### 6.18.3.9 void MadGuid::setGuid ( QString *theGuid* = " " ) [inherited]

MadGuid::setGuid.

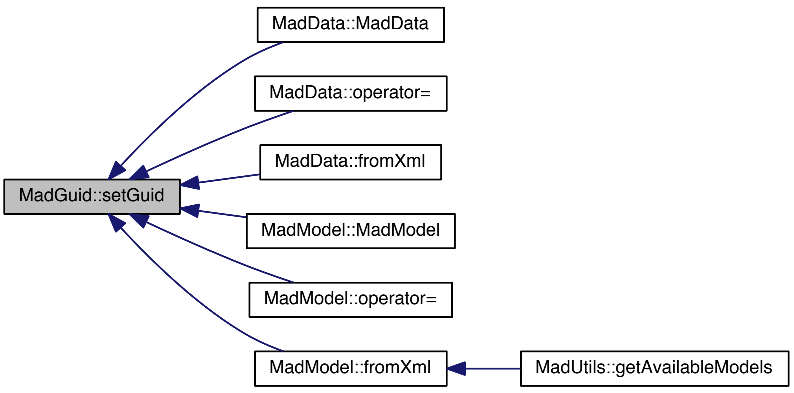
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:

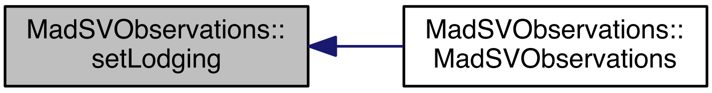


#### 6.18.3.10 void MadSVObservations::setLodging ( MadSubCategory *theData* )

Definition at line 72 of file madsvobservations.cpp.

```
{
  mLodging = theData;
}
```

Here is the caller graph for this function:

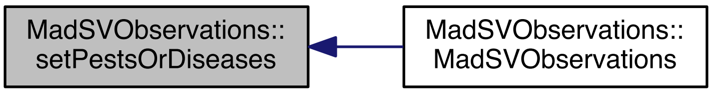


#### 6.18.3.11 void MadSVObservations::setPestsOrDiseases ( MadSubCategory *theData* )

Definition at line 77 of file madsvobservations.cpp.

```
{
  mPestsOrDiseases = theData;
}
```

Here is the caller graph for this function:



#### 6.18.3.12 QString MadSVObservations::toHtml ( )

Return a html text representation of this layer

Definition at line 137 of file madsvobservations.cpp.

```
{
  QString myString;
  //myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  //myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";

  //
  // the following shows example of how to do a couple of things
  //

  //myString+="<tr><td><b>Cals/Kg: </b></td><td>" + QString::number(mCropCalories) + "</td></tr>";
  //QString myCropFodderEnergyType = (mCropFodderEnergyType==0) ? "KCalories" : "TDN";
  //QString myUnits = (mAreaUnits==0) ? "Dunum" : "Hectare";
  //myString+="<tr><td><b>Fodder (kg/" + myUnits + "): </b></td><td>" +
   QString::number(mCropFodderProduction) + "</td></tr>";
  //myString+="<tr><td><b>Fodder Value/Kg: </b></td><td>" + QString::number(mCropFodderValue) +
   "</td></tr>";
  //myString+="<tr><td><b>FodderEnergyType: </b></td><td>" + myCropFodderEnergyType + "</td></tr>";
  //myString+="<tr><td><b>AreaUnits: </b></td><td>" + myUnits + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

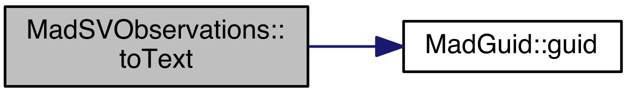
### 6.18.3.13 QString MadSVObservations::toText ( )

Return a plain text representation of this layer

Definition at line 128 of file madsvobservations.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  //myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



### 6.18.3.14 QString MadSVObservations::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

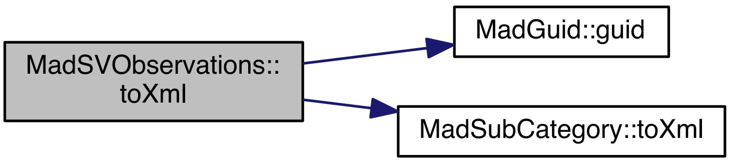
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

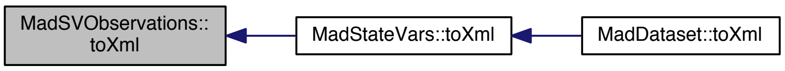
Definition at line 106 of file madsvobservations.cpp.

```
{
  QString myString;
  myString+=QString("    <svobservations guid=\"" + guid() + "\">\n");

  myString+=QString("      <lodging>\n");
  myString+=mLodging.toXml();
  myString+=QString("      </lodging>\n");

  myString+=QString("      <pestsordiseases>\n");
  myString+=mPestsOrDiseases.toXml();
  myString+=QString("      </pestsordiseases>\n");

  myString+=QString("      <damage>\n");
  myString+=mDamages.toXml();
  myString+=QString("      </damage>\n");

  mySString+=QString("    </svobservations>\n");
  return myString;

}
```

Here is the call graph for this function:



Here is the caller graph for this function:



#### 6.18.3.15 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

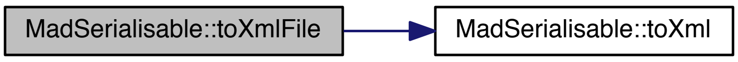
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



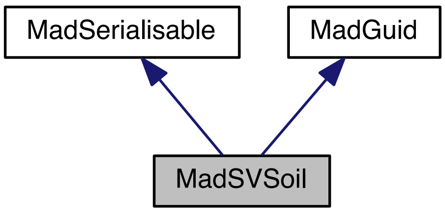
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madsvobservations.-h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madsvobservations.-cpp

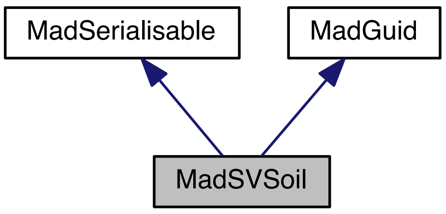
## 6.19 MadSVSoil Class Reference

`#include <madsvsoil.h>`

Inheritance diagram for MadSVSoil:



Collaboration diagram for MadSVSoil:



### Public Member Functions

- MadSVSoil ()
- MadSVSoil (const MadSVSoil &theData)
- MadSVSoil & operator= (const MadSVSoil &theData)
- MadSubCategory soilWaterGrav () const
- MadSubCategory pressureHeads () const
- MadSubCategory nMin () const
- MadSubCategory soilWaterSensorCal () const
- MadSubCategory waterFluxBottomRoot () const
- MadSubCategory nitrogenFluxBottomRoot () const
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)
- void setSoilWaterGrav (MadSubCategory theData)
- void setPressureHeads (MadSubCategory theData)
- void setNMin (MadSubCategory theData)
- void setSoilWaterSensorCal (MadSubCategory theData)
- void setWaterFluxBottomRoot (MadSubCategory theData)
- void setNitrogenFluxBottomRoot (MadSubCategory theData)
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.19.1 Detailed Description

Definition at line 35 of file madsvsoil.h.

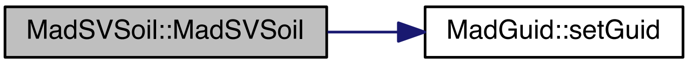
### 6.19.2 Constructor & Destructor Documentation

#### 6.19.2.1 MadSVSoil::MadSVSoil ( )

Definition at line 34 of file madsvsoil.cpp.

```
                       : MadSerialisable(), MadGuid()
{
  setGuid();
}
```

Here is the call graph for this function:

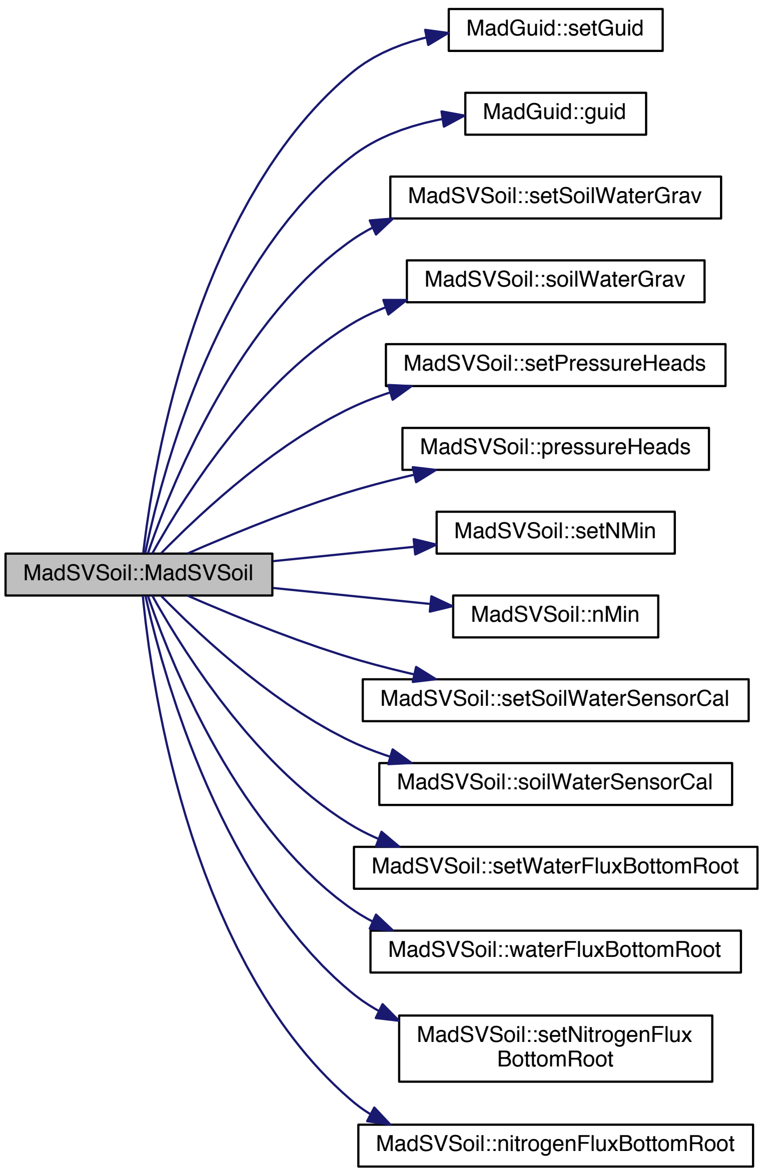


#### 6.19.2.2 MadSVSoil::MadSVSoil ( const MadSVSoil & *theData* )

Definition at line 39 of file madsvsoil.cpp.

```
{
  setGuid(theData.guid());
  setSoilWaterGrav(theData.soilWaterGrav());
  setPressureHeads(theData.pressureHeads());
  setNMin(theData.nMin());
  setSoilWaterSensorCal(theData.soilWaterSensorCal());
  setWaterFluxBottomRoot(theData.waterFluxBottomRoot());
  setNitrogenFluxBottomRoot(theData.
   nitrogenFluxBottomRoot());
}
```

Here is the call graph for this function:


MadGuid::setGuid
MadGuid::guid
MadSVSoil::setSoilWaterGrav
MadSVSoil::soilWaterGrav
MadSVSoil::setPressureHeads
MadSVSoil::pressureHeads
MadSVSoil::setNMin
MadSVSoil::MadSVSoil
MadSVSoil::nMin
MadSVSoil::setSoilWaterSensorCal
MadSVSoil::soilWaterSensorCal
MadSVSoil::setWaterFluxBottomRoot
MadSVSoil::waterFluxBottomRoot
MadSVSoil::setNitrogenFlux BottomRoot
MadSVSoil::nitrogenFluxBottomRoot

### 6.19.3 Member Function Documentation

#### 6.19.3.1 bool MadSVSoil::fromXml ( const QString *theXml* ) `[virtual]`

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

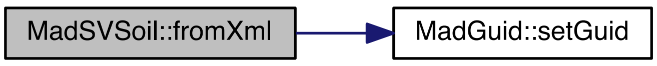
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 127 of file madsvsoil.cpp.

```
{
    QDomDocument myDocument("mydocument");
    myDocument.setContent(theXml);
    QDomElement myTopElement = myDocument.firstChildElement("svsoil");
    if (myTopElement.isNull())
    {
        //TODO - just make this a warning
        qDebug("the top element couldn't be found!");
        setGuid(myTopElement.attribute("guid"));
        //mName=MadUtils::xmlDecode(myTopElement.firstChildElement("name").text());
        //mDescription=MadUtils::xmlDecode(myTopElement.firstChildElement("description").text());
        //mImageFile=QString(myTopElement.firstChildElement("imageFile").text());
        return true;
    }
    else
    return false;
}
```

Here is the call graph for this function:



#### 6.19.3.2 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) `[virtual], [inherited]`

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

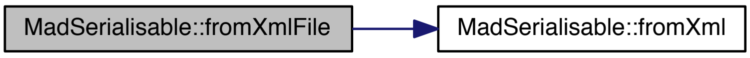
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



### 6.19.3.3 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

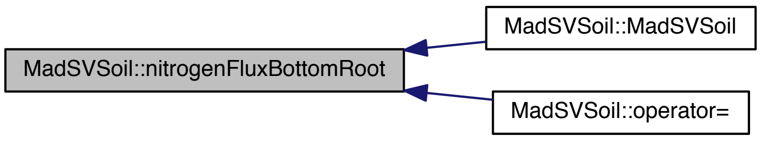
```
{
    return mGuid;
}
```

### 6.19.3.4 MadSubCategory MadSVSoil::nitrogenFluxBottomRoot ( ) const

Definition at line 91 of file madsvsoil.cpp.

```
{
  return mNitrogenFluxBottomRoot;
}
```

Here is the caller graph for this function:

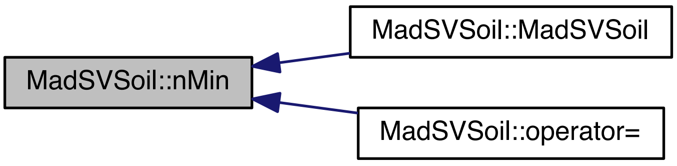


#### 6.19.3.5 MadSubCategory MadSVSoil::nMin ( ) const

Definition at line 76 of file madsvsoil.cpp.

```
{
  return mNMin;
}
```

Here is the caller graph for this function:

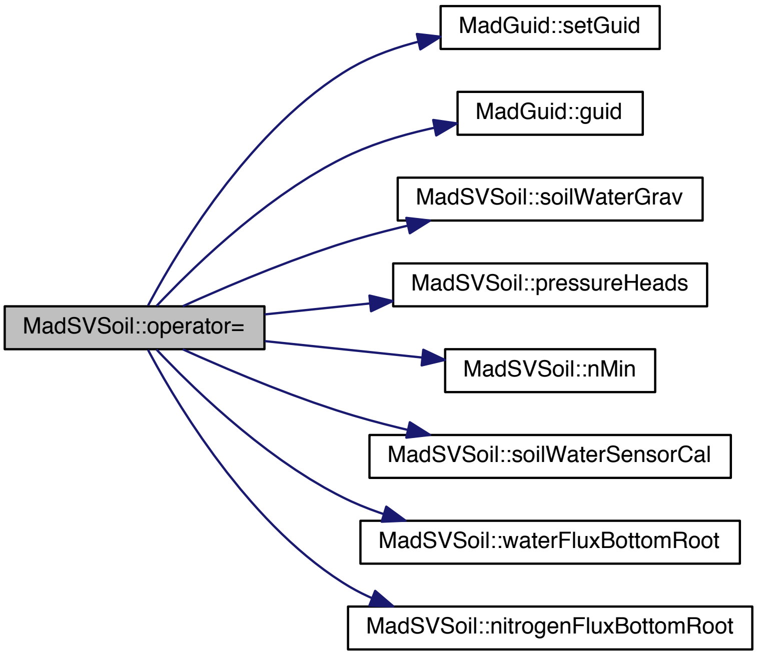


#### 6.19.3.6 MadSVSoil & MadSVSoil::operator= ( const MadSVSoil & *theData* )

Definition at line 50 of file madsvsoil.cpp.

```
{
  // gracefully handles self assignment
  if (this == &theData) return *this;
  setGuid(theData.guid());
  mSoilWaterGrav=theData.soilWaterGrav();
  mPressureHeads=theData.pressureHeads();
  mNMin=theData.nMin();
  mSoilWaterSensorCal=theData.soilWaterSensorCal();
  mWaterFluxBottomRoot=theData.waterFluxBottomRoot();
  mNitrogenFluxBottomRoot=theData.nitrogenFluxBottomRoot();
  return *this;
}
```

Here is the call graph for this function:

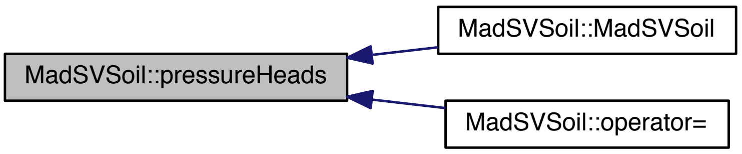


#### 6.19.3.7 MadSubCategory MadSVSoil::pressureHeads ( ) const

Definition at line 71 of file madsvsoil.cpp.

```
72 {
73   return mPressureHeads;
74 }
```

Here is the caller graph for this function:



#### 6.19.3.8 void MadGuid::setGuid ( QString *theGuid* = " " ) [inherited]

MadGuid::setGuid.

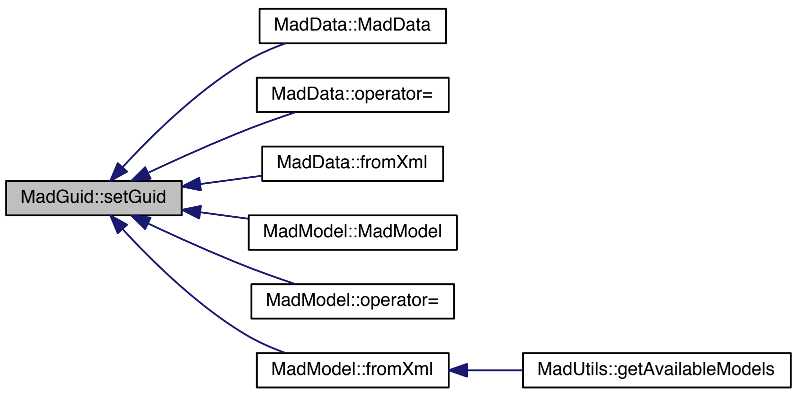
**Parameters**

|  |  |
|---|---|
| *theGuid* |  |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:

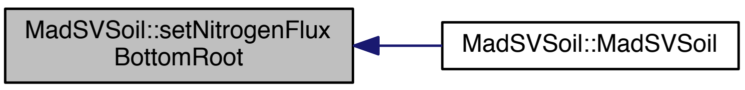


#### 6.19.3.9 void MadSVSoil::setNitrogenFluxBottomRoot ( MadSubCategory *theData* )

Definition at line 122 of file madsvsoil.cpp.

```
{
  mNitrogenFluxBottomRoot = theData;
}
```

Here is the caller graph for this function:

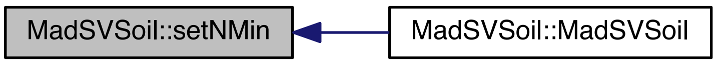


#### 6.19.3.10 void MadSVSoil::setNMin ( MadSubCategory *theData* )

Definition at line 107 of file madsvsoil.cpp.

```
{
  mNMin = theData;
}
```

Here is the caller graph for this function:

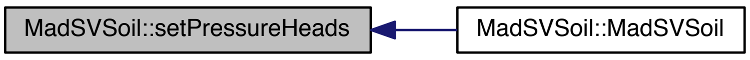


#### 6.19.3.11 void MadSVSoil::setPressureHeads ( MadSubCategory *theData* )

Definition at line 102 of file madsvsoil.cpp.

```
{
  mPressureHeads = theData;
}
```

Here is the caller graph for this function:

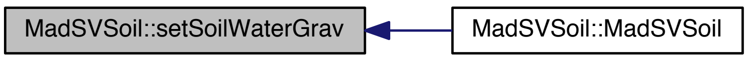


#### 6.19.3.12 void MadSVSoil::setSoilWaterGrav ( MadSubCategory *theData* )

Definition at line 97 of file madsvsoil.cpp.

```
{
  mSoilWaterGrav = theData;
}
```

Here is the caller graph for this function:

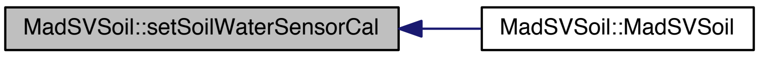


#### 6.19.3.13 void MadSVSoil::setSoilWaterSensorCal ( MadSubCategory *theData* )

Definition at line 112 of file madsvsoil.cpp.

```
{
  mSoilWaterSensorCal = theData;
}
```

Here is the caller graph for this function:

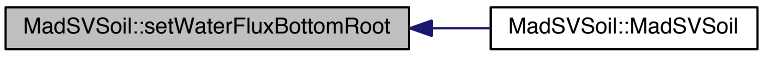


#### 6.19.3.14 void MadSVSoil::setWaterFluxBottomRoot ( MadSubCategory *theData* )

Definition at line 117 of file madsvsoil.cpp.

```
{
  mWaterFluxBottomRoot = theData;
}
```

Here is the caller graph for this function:

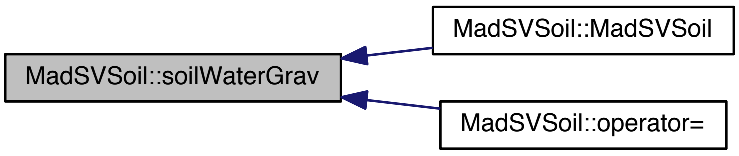


#### 6.19.3.15 MadSubCategory MadSVSoil::soilWaterGrav ( ) const

Definition at line 66 of file madsvsoil.cpp.

```
{
  return mSoilWaterGrav;
}
```

Here is the caller graph for this function:

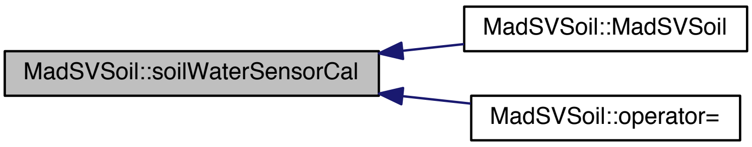


#### 6.19.3.16 MadSubCategory MadSVSoil::soilWaterSensorCal ( ) const

Definition at line 81 of file madsvsoil.cpp.

```
{
  return mSoilWaterSensorCal;
}
```

Here is the caller graph for this function:



#### 6.19.3.17 QString MadSVSoil::toHtml ( )

Return a html text representation of this layer

Definition at line 190 of file madsvsoil.cpp.

```
{
  QString myString;
  //myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  //myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";

  //
  // the following shows example of how to do a couple of things
  //

  //myString+="<tr><td><b>Cals/Kg: </b></td><td>" + QString::number(mCropCalories) + "</td></tr>";
  //QString myCropFodderEnergyType = (mCropFodderEnergyType==0) ? "KCalories" : "TDN";
  //QString myUnits = (mAreaUnits==0) ? "Dunum" : "Hectare";
  //myString+="<tr><td><b>Fodder (kg/" + myUnits + "): </b></td><td>" +
   QString::number(mCropFodderProduction) + "</td></tr>";
  //myString+="<tr><td><b>Fodder Value/Kg: </b></td><td>" + QString::number(mCropFodderValue) +
   "</td></tr>";
  //myString+="<tr><td><b>FodderEnergyType: </b></td><td>" + myCropFodderEnergyType + "</td></tr>";
  //myString+="<tr><td><b>AreaUnits: </b></td><td>" + myUnits + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

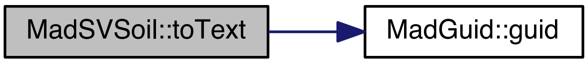
#### 6.19.3.18 QString MadSVSoil::toText ( )

Return a plain text representation of this layer

Definition at line 181 of file madsvsoil.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("name=>" + MadUtils::xmlEncode(mName) + "\n");
  //myString+=QString("description=>" + MadUtils::xmlEncode(mDescription) + "\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.19.3.19 QString MadSVSoil::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

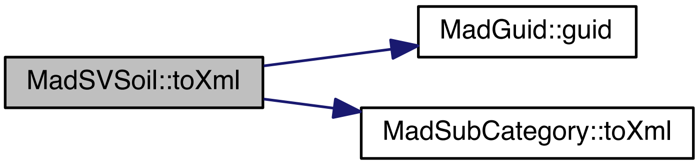
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

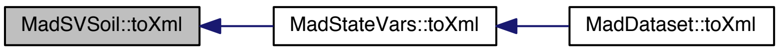
Definition at line 146 of file madsvsoil.cpp.

```
{
  QString myString;
  myString+=QString("    <svsoil guid=\"" + guid() + "\">\n");

  myString+=QString("     <soilwatergrav>\n");
  myString+=mSoilWaterGrav.toXml();
  myString+=QString("     </soilwatergrav>\n");

  myString+=QString("     <pressureheads>\n");
  myString+=mPressureHeads.toXml();
  myString+=QString("     </pressureheads>\n");

  myString+=QString("     <nmin>\n");
  myString+=mNMin.toXml();
  myString+=QString("     </nmin>\n");

  myString+=QString("     <soilwatersensorcal>\n");
  myString+=mSoilWaterSensorCal.toXml();
  myString+=QString("     </soilwatersensorcal>\n");

  myString+=QString("     <waterfluxbottomroot>\n");
  myString+=mWaterFluxBottomRoot.toXml();
  myString+=QString("     </waterfluxbottomroot>\n");

  myString+=QString("     <nitrogenfluxbottomroot>\n");
  myString+=mNitrogenFluxBottomRoot.toXml();
  myString+=QString("     </nitrogenfluxbottomroot>\n");

  myString+=QString("    </svsoil>\n");
  return myString;


}
```

Here is the call graph for this function:



Here is the caller graph for this function:



### 6.19.3.20 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

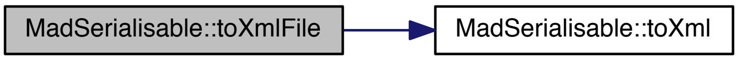
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:

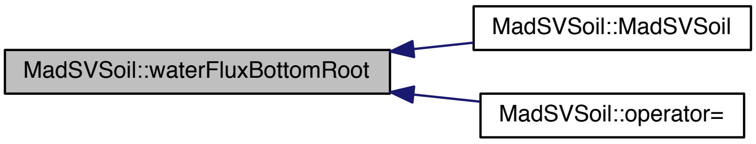


### 6.19.3.21 MadSubCategory MadSVSoil::waterFluxBottomRoot ( ) const

Definition at line 86 of file madsvsoil.cpp.

```
{
  return mWaterFluxBottomRoot;
}
```

Here is the caller graph for this function:



The documentation for this class was generated from the following files:

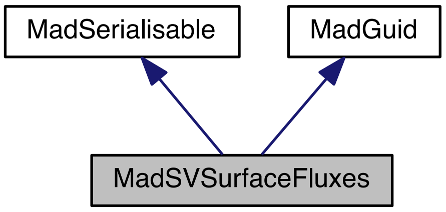
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madsvsoil.h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madsvsoil.-cpp

## 6.20 MadSVSurfaceFluxes Class Reference

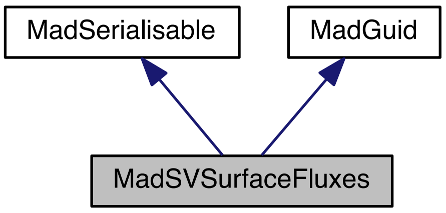
The MadSVSurfaceFluxes class.

```
#include <madsvsurfacefluxes.h>
```

Inheritance diagram for MadSVSurfaceFluxes:



Collaboration diagram for MadSVSurfaceFluxes:



## Public Member Functions

- MadSVSurfaceFluxes ()
- MadSVSurfaceFluxes (const MadSVSurfaceFluxes &theData)
- MadSVSurfaceFluxes & operator= (const MadSVSurfaceFluxes &theData)
- MadSubCategory et () const

  *et*
- MadSubCategory nh3Loss () const

  *nh3Loss*
- MadSubCategory n2oLoss () const

  *n2oLoss*
- MadSubCategory n2Loss () const

  *n2Loss*
- MadSubCategory ch4Loss () const

  *ch4Loss*
- QString toXml ()
- QString toText ()
- QString toHtml ()
- bool fromXml (const QString theXml)

- void setEt (MadSubCategory theData)

  *setEt*
- void setNh3Loss (MadSubCategory theData)

  *setNh3Loss*
- void setN2oLoss (MadSubCategory theData)

  *setN2oLoss*
- void setN2Loss (MadSubCategory theData)

  *setN2Loss*
- void setCh4Loss (MadSubCategory theData)

  *setCh4Loss*
- virtual bool toXmlFile (const QString theFileName)

  *toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.*
- virtual bool fromXmlFile (const QString theFileName)

  *fromXmlFile Read this object from xml in a file*
- QString guid () const

  *MadGuid::guid.*
- void setGuid (QString theGuid="")

  *MadGuid::setGuid.*

### 6.20.1 Detailed Description

The MadSVSurfaceFluxes class.

Definition at line 38 of file madsvsurfacefluxes.h.

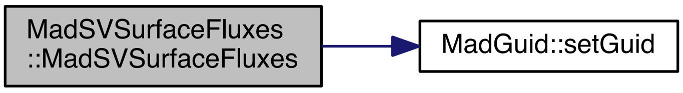
### 6.20.2 Constructor & Destructor Documentation

#### 6.20.2.1 MadSVSurfaceFluxes::MadSVSurfaceFluxes ( )

Definition at line 32 of file madsvsurfacefluxes.cpp.

```
                                          : MadSerialisable(),
     MadGuid()
{
  setGuid();

  MadSubCategory mEt;
  MadSubCategory mNh3Loss;
  MadSubCategory mN2oLoss;
  MadSubCategory mN2Loss;
  MadSubCategory mCh4Loss;
}
```

Here is the call graph for this function:

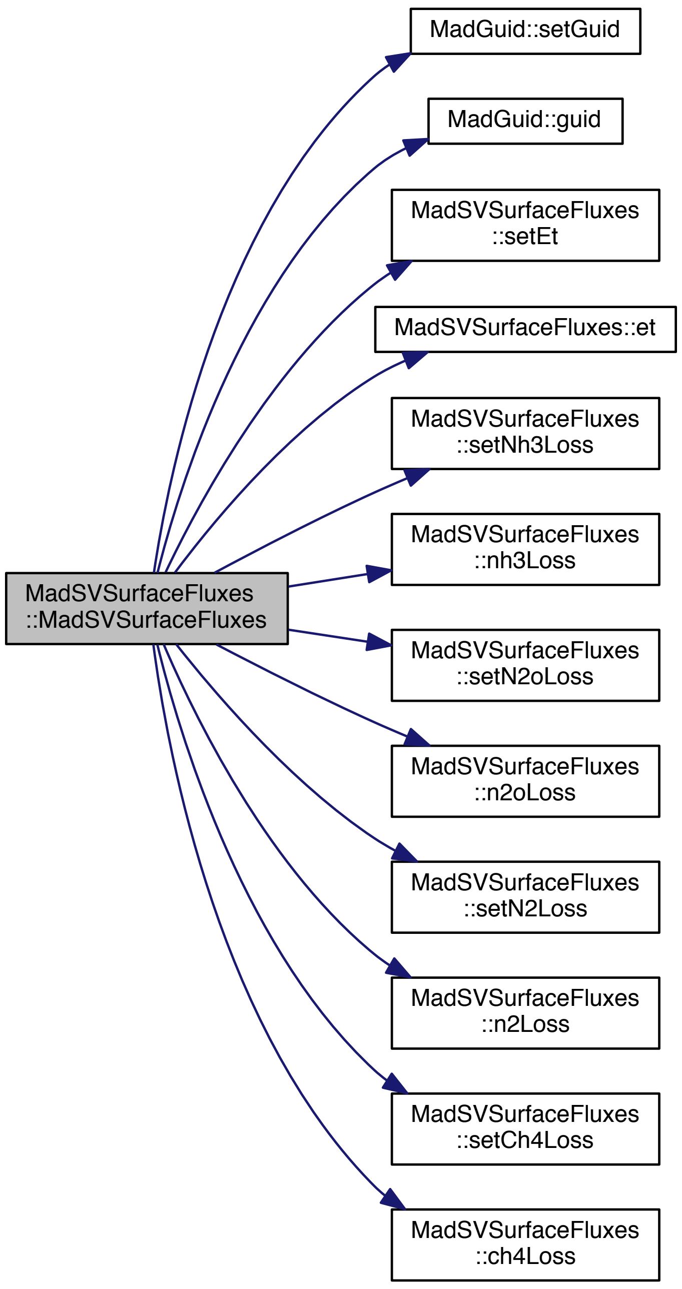


#### 6.20.2.2 MadSVSurfaceFluxes::MadSVSurfaceFluxes ( const MadSVSurfaceFluxes & *theData* )

Definition at line 43 of file madsvsurfacefluxes.cpp.

```
{
  setGuid(theData.guid());
  setEt(theData.et());
  setNh3Loss(theData.nh3Loss());
  setN2oLoss(theData.n2oLoss());
  setN2Loss(theData.n2Loss());
  setCh4Loss(theData.ch4Loss());
}
```

Here is the call graph for this function:


MadGuid::setGuid
MadGuid::guid
MadSVSurfaceFluxes ::setEt
MadSVSurfaceFluxes::et
MadSVSurfaceFluxes ::setNh3Loss
MadSVSurfaceFluxes ::nh3Loss
MadSVSurfaceFluxes ::MadSVSurfaceFluxes
MadSVSurfaceFluxes ::setN2oLoss
MadSVSurfaceFluxes ::n2oLoss
MadSVSurfaceFluxes ::setN2Loss
MadSVSurfaceFluxes ::n2Loss
MadSVSurfaceFluxes ::setCh4Loss
MadSVSurfaceFluxes ::ch4Loss

### 6.20.3 Member Function Documentation

#### 6.20.3.1 MadSubCategory MadSVSurfaceFluxes::ch4Loss ( ) const

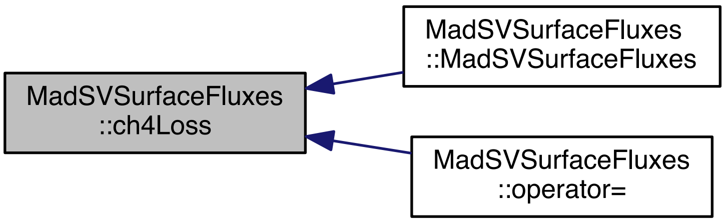
ch4Loss

**Returns**

Definition at line 86 of file madsvsurfacefluxes.cpp.

```
{
  return mCh4Loss;
}
```

Here is the caller graph for this function:


MadSVSurfaceFluxes ::MadSVSurfaceFluxes
MadSVSurfaceFluxes ::ch4Loss
MadSVSurfaceFluxes ::operator=

#### 6.20.3.2 MadSubCategory MadSVSurfaceFluxes::et ( ) const

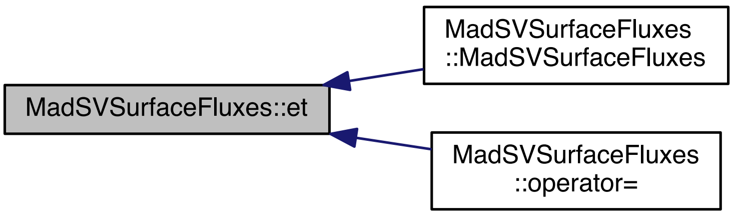
et

**Returns**

Definition at line 67 of file madsvsurfacefluxes.cpp.

```
{
  return mEt;
}
```

Here is the caller graph for this function:


MadSVSurfaceFluxes ::MadSVSurfaceFluxes
MadSVSurfaceFluxes::et
MadSVSurfaceFluxes ::operator=

#### 6.20.3.3 bool MadSVSurfaceFluxes::fromXml ( const QString *theXml* ) [virtual]

Read this object from xml and return result as true for success, false for failure.

**See Also**

MadSerialisable

**Note**

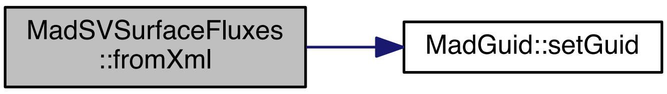
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

Definition at line 117 of file madsvsurfacefluxes.cpp.

```
{
    QDomDocument myDocument("mydocument");
    myDocument.setContent(theXml);
    QDomElement myTopElement = myDocument.firstChildElement("surfacefluxes");
    if (myTopElement.isNull())
    {
        //TODO - just make this a warning
        qDebug("the top element couldn't be found!");
        setGuid(myTopElement.attribute("guid"));
        //mEt=MadUtils::xmlDecode(myTopElement.firstChildElement("et").text());
        //mNh3Loss=MadUtils::xmlDecode(myTopElement.firstChildElement("nh3Loss").text());
        //mN2oLoss=MadUtils::xmlDecode(myTopElement.firstChildElement("n2oLoss").text());
        //mN2Loss=MadUtils::xmlDecode(myTopElement.firstChildElement("n2Loss").text());
        //ch4Loss()=MadUtils::xmlDecode(myTopElement.firstChildElement("ch4Loss").text());
        return true;
    }
    else
    return false;
}
```

Here is the call graph for this function:



#### 6.20.3.4 bool MadSerialisable::fromXmlFile ( const QString *theFileName* ) [virtual], [inherited]

fromXmlFile Read this object from xml in a file

**See Also**

fromXmlFile() Internally it uses fromXml(QString) so that must be properly implemented

**Parameters**

| | |
|---|---|
| *theFileName* | |

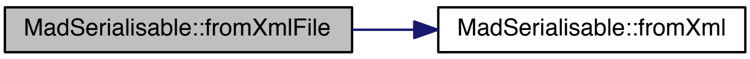
**Returns**

result as true for success, false for failure.

Definition at line 76 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::ReadOnly ) )
  {
    myResult=this->fromXml(myFile.readAll());
    myFile.close();
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



### 6.20.3.5 QString MadGuid::guid ( ) const [inherited]

MadGuid::guid.

Destructor Retrieve the GUID

**Returns**

Definition at line 40 of file madguid.cpp.

```
{
    return mGuid;
}
```

### 6.20.3.6 MadSubCategory MadSVSurfaceFluxes::n2Loss ( ) const

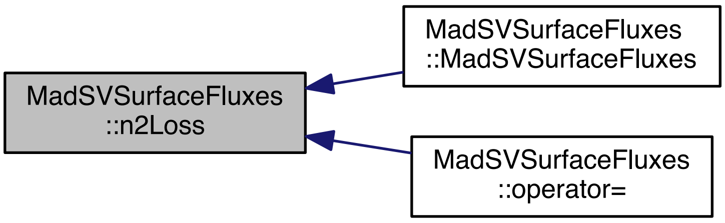
n2Loss

**Returns**

Definition at line 81 of file madsvsurfacefluxes.cpp.

```
{
  return mN2Loss;
}
```

Here is the caller graph for this function:


MadSVSurfaceFluxes ::n2Loss
MadSVSurfaceFluxes ::MadSVSurfaceFluxes
MadSVSurfaceFluxes ::operator=

#### 6.20.3.7 MadSubCategory MadSVSurfaceFluxes::n2oLoss ( ) const

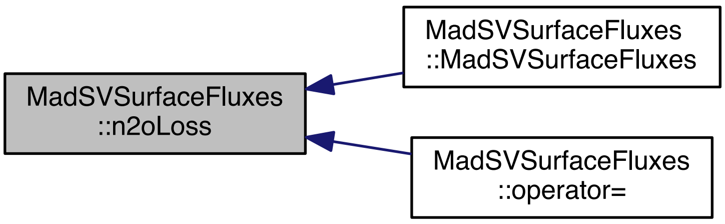
n2oLoss

**Returns**

Definition at line 76 of file madsvsurfacefluxes.cpp.

```
{
  return mN2oLoss;
}
```

Here is the caller graph for this function:


MadSVSurfaceFluxes ::n2oLoss
MadSVSurfaceFluxes ::MadSVSurfaceFluxes
MadSVSurfaceFluxes ::operator=

#### 6.20.3.8 MadSubCategory MadSVSurfaceFluxes::nh3Loss ( ) const

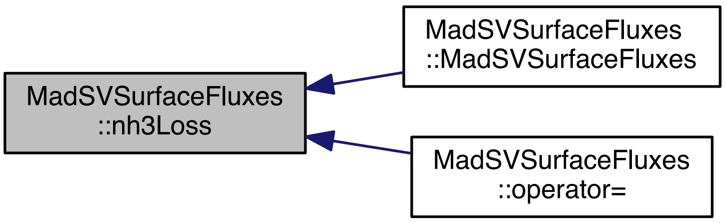
nh3Loss

**Returns**

Definition at line 71 of file madsvsurfacefluxes.cpp.

```
{
  return mNh3Loss;
}
```

Here is the caller graph for this function:

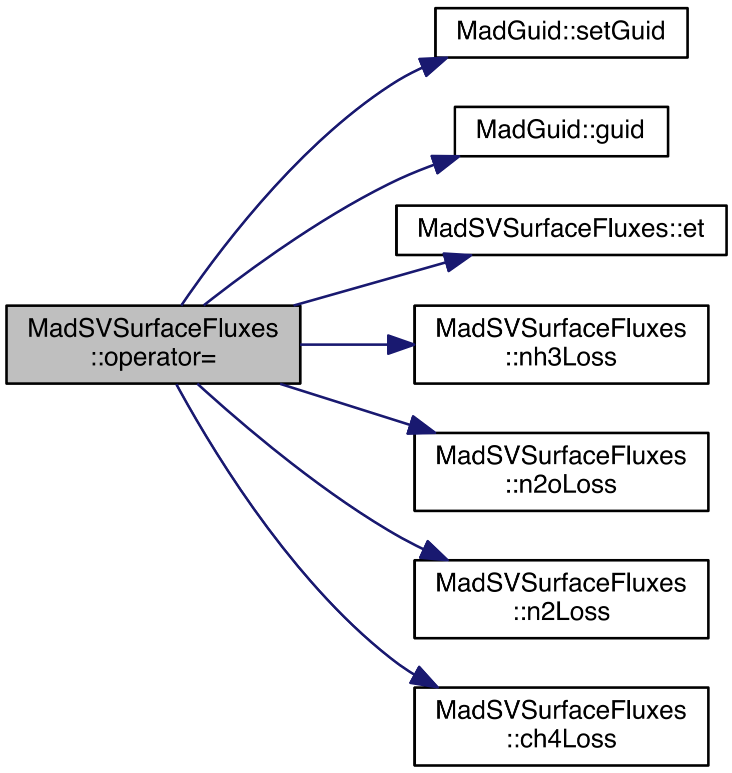

MadSVSurfaceFluxes ::nh3Loss
MadSVSurfaceFluxes ::MadSVSurfaceFluxes
MadSVSurfaceFluxes ::operator=

#### 6.20.3.9 **MadSVSurfaceFluxes** & MadSVSurfaceFluxes::operator= ( const **MadSVSurfaceFluxes** & *theData* )

Definition at line 53 of file madsvsurfacefluxes.cpp.

```
{
  // gracefully handles self assignment
  if (this == &theData) return *this;
  setGuid(theData.guid());
  mEt=theData.et();
  mNh3Loss=theData.nh3Loss();
  mN2oLoss=theData.n2oLoss();
  mN2Loss=theData.n2Loss();
  mCh4Loss=theData.ch4Loss();
  return *this;
}
```

Here is the call graph for this function:


MadSVSurfaceFluxes ::operator=
MadGuid::setGuid
MadGuid::guid
MadSVSurfaceFluxes::et
MadSVSurfaceFluxes ::nh3Loss
MadSVSurfaceFluxes ::n2oLoss
MadSVSurfaceFluxes ::n2Loss
MadSVSurfaceFluxes ::ch4Loss

#### 6.20.3.10 void MadSVSurfaceFluxes::setCh4Loss ( **MadSubCategory** *theData* )

setCh4Loss

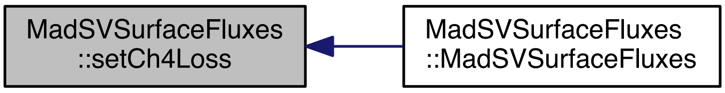
**Parameters**

| | |
|---|---|
| *theData* | |

Definition at line 112 of file madsvsurfacefluxes.cpp.

```
{
  mCh4Loss = theData;
}
```

Here is the caller graph for this function:


MadSVSurfaceFluxes ::setCh4Loss
MadSVSurfaceFluxes ::MadSVSurfaceFluxes

#### 6.20.3.11 void MadSVSurfaceFluxes::setEt ( **MadSubCategory** *theData* )

setEt

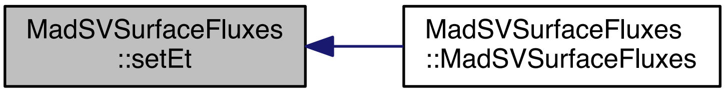
**Parameters**

| | |
|---|---|
| *theData* | |

Definition at line 92 of file madsvsurfacefluxes.cpp.

```
{
  mEt = theData;
}
```

Here is the caller graph for this function:


MadSVSurfaceFluxes ::setEt
MadSVSurfaceFluxes ::MadSVSurfaceFluxes

#### 6.20.3.12 void MadGuid::setGuid ( QString *theGuid =* " " ) [inherited]

MadGuid::setGuid.

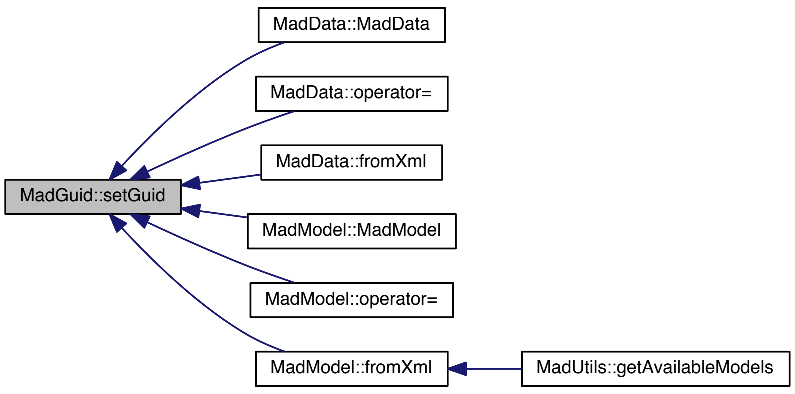
**Parameters**

| | |
|---|---|
| *theGuid* | |

Definition at line 49 of file madguid.cpp.

```
{
    if (theGuid.isEmpty())
    {
        mGuid=QUuid::createUuid().toString().replace("{","").replace("}","");
    }
    else
    {
        mGuid=theGuid;
    }
}
```

Here is the caller graph for this function:



#### 6.20.3.13 void MadSVSurfaceFluxes::setN2Loss ( MadSubCategory *theData* )

setN2Loss

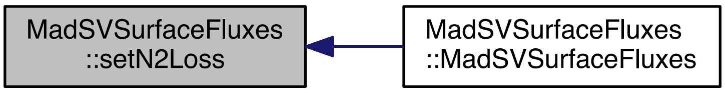
**Parameters**

| | |
|---|---|
| *theData* | |

Definition at line 107 of file madsvsurfacefluxes.cpp.

```
{
  mN2Loss = theData;
}
```

Here is the caller graph for this function:


MadSVSurfaceFluxes ::setN2Loss
MadSVSurfaceFluxes ::MadSVSurfaceFluxes

#### 6.20.3.14 void MadSVSurfaceFluxes::setN2oLoss ( MadSubCategory *theData* )

setN2oLoss

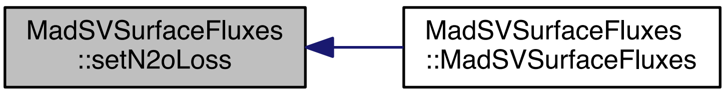
**Parameters**

| | |
|---|---|
| *theData* | |

Definition at line 102 of file madsvsurfacefluxes.cpp.

```
{
  mN2oLoss = theData;
}
```

Here is the caller graph for this function:


MadSVSurfaceFluxes ::setN2oLoss
MadSVSurfaceFluxes ::MadSVSurfaceFluxes

### 6.20.3.15 void MadSVSurfaceFluxes::setNh3Loss ( MadSubCategory *theData* )

setNh3Loss

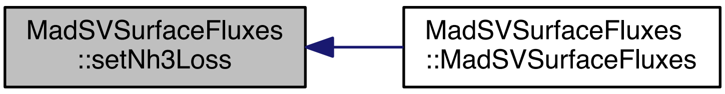
**Parameters**

| | |
|---|---|
| *theData* | |

Definition at line 97 of file madsvsurfacefluxes.cpp.

```
{
  mNh3Loss = theData;
}
```

Here is the caller graph for this function:


MadSVSurfaceFluxes ::setNh3Loss
MadSVSurfaceFluxes ::MadSVSurfaceFluxes

### 6.20.3.16 QString MadSVSurfaceFluxes::toHtml ( )

Return a html text representation of this layer

Definition at line 180 of file madsvsurfacefluxes.cpp.

```
{
  QString myString;
  //myString+="<h3>Details for " + MadUtils::xmlEncode(mName) + "</h3>";
    //myString+="<p>GUID:" + guid() + "</p>";
  myString+="<table>";
  //myString+="<tr><td><b>Description: </b></td><td>" + mDescription + "</td></tr>";

  //
  // the following shows example of how to do a couple of things
  //

  //myString+="<tr><td><b>Cals/Kg: </b></td><td>" + QString::number(mCropCalories) + "</td></tr>";
  //QString myCropFodderEnergyType = (mCropFodderEnergyType==0) ? "KCalories" : "TDN";
  //QString myUnits = (mAreaUnits==0) ? "Dunum" : "Hectare";
  //myString+="<tr><td><b>Fodder (kg/" + myUnits + "): </b></td><td>" +
   QString::number(mCropFodderProduction) + "</td></tr>";
  //myString+="<tr><td><b>Fodder Value/Kg: </b></td><td>" + QString::number(mCropFodderValue) +
   "</td></tr>";
  //myString+="<tr><td><b>FodderEnergyType: </b></td><td>" + myCropFodderEnergyType + "</td></tr>";
  //myString+="<tr><td><b>AreaUnits: </b></td><td>" + myUnits + "</td></tr>";
  myString+="</table>";
  return myString;
}
```

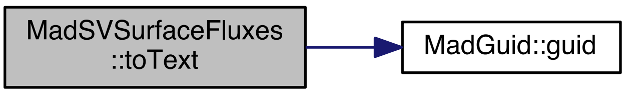
#### 6.20.3.17 QString MadSVSurfaceFluxes::toText ( )

Return a plain text representation of this layer I need to figure out how to turn the sub category into text

Definition at line 167 of file madsvsurfacefluxes.cpp.

```
{
  QString myString;
  myString+=QString("guid=>" + guid() + "\n");
  //myString+=QString("et=>" + MadUtils::xmlEncode(mEt) + "</et>\n");
  //myString+=QString("nh3Loss=>" + MadUtils::xmlEncode(mNh3Loss) + "</nh3Loss>\n");
  //myString+=QString("n2oLoss=>" + MadUtils::xmlEncode(mN2oLoss) + "</n2oLoss>\n");
  //myString+=QString("n2Loss=>" + MadUtils::xmlEncode(mN2Loss) + "</n2Loss>\n");
  //myString+=QString("ch4Loss=>" + MadUtils::xmlEncode(mCh4Loss) + "</ch4Loss>\n");
  return myString;
}
```

Here is the call graph for this function:



#### 6.20.3.18 QString MadSVSurfaceFluxes::toXml ( ) [virtual]

Return an xml representation of this layer

**Note**

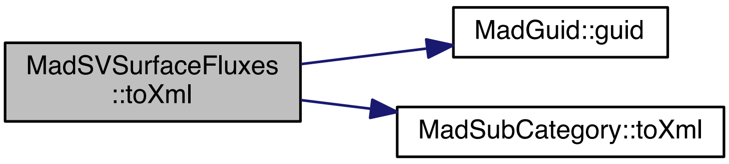
this class inherits the serialisable interface so it MUST implement this

Implements MadSerialisable.

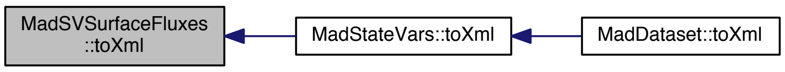
Definition at line 138 of file madsvsurfacefluxes.cpp.

```
{
  QString myString;
  myString+=QString("     <surfacefluxes guid=\"" + guid() + "\">\n");

  myString+=QString("      <et>\n");
  myString+=mEt.toXml();
  myString+=QString("      </et>\n");

  myString+=QString("      <nh3loss>");
  myString+=mNh3Loss.toXml();
  myString+=QString("      </nh3loss>\n");

  myString+=QString("      <n2oloss>");
  myString+=mN2oLoss.toXml();
  myString+=QString("      </n2oloss>\n");

  myString+=QString("      <n2loss>");
  myString+=mN2Loss.toXml();
  myString+=QString("      </n2loss>\n");

  myString+=QString("      <ch4loss>");
  myString+=mCh4Loss.toXml();
  myString+=QString("      </ch4loss>\n");

  myString+=QString("     </surfacefluxes>\n");
  return myString;
}
```

Here is the call graph for this function:



Here is the caller graph for this function:



#### 6.20.3.19 bool MadSerialisable::toXmlFile ( const QString *theFileName* ) [virtual], [inherited]

toXmlFile writes object to xml and return result (virtual qstring) We provide a basic default implementation where given a file name, we will write the serialised xml to that file. Internally it uses toXml() method so that must be properly implemented.

**See Also**

toXml()

**Parameters**

| | |
|---|---|
| *theFileName* | |

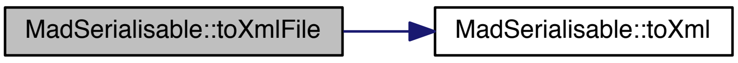
**Returns**

QString (virtual)

Definition at line 57 of file madserialisable.cpp.

```
{
  bool myResult = false;
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << this->toXml();
    myFile.close();
    myResult=true;
  }
  else
  {
    //@TODO Error handler!
    myResult=false;
  }
  return myResult ;
}
```

Here is the call graph for this function:



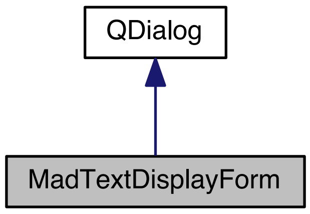
The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madsvsurfacefluxes.-h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madsvsurfacefluxes.-cpp

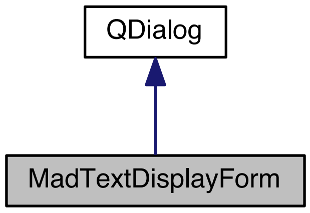
## 6.21 MadTextDisplayForm Class Reference

```
#include <madtextdisplayform.h>
```

Inheritance diagram for MadTextDisplayForm:



Collaboration diagram for MadTextDisplayForm:



## Public Member Functions

- MadTextDisplayForm (QWidget ∗parent=0)
- ∼MadTextDisplayForm ()
- void setText (const QString &theText)

### 6.21.1 Detailed Description

Definition at line 39 of file madtextdisplayform.h.

### 6.21.2 Constructor & Destructor Documentation

#### 6.21.2.1 MadTextDisplayForm::MadTextDisplayForm ( QWidget ∗ *parent* = 0 ) [explicit]

Definition at line 25 of file madtextdisplayform.cpp.

```
                                                        :
  QDialog(parent),
  ui(new Ui::MadTextDisplayForm)
{
  ui->setupUi(this);
}
```

#### 6.21.2.2 MadTextDisplayForm::∼MadTextDisplayForm ( )

Definition at line 32 of file madtextdisplayform.cpp.

```
{
  delete ui;
}
```

### 6.21.3 Member Function Documentation

#### 6.21.3.1 void MadTextDisplayForm::setText ( const QString & *theText* )

Definition at line 37 of file madtextdisplayform.cpp.

```
{
  ui->textBrowser->setText(theText);
}
```

The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/madtextdisplayform.h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/madtextdisplayform.cpp

## 6.22 MadUtils Class Reference

```
#include <madutils.h>
```

### Public Types

- typedef QMap< QString, MadModel > ModelMap

  *ModelMap (typedef) This typedef is used to refer to a collection of layersets. the key is the layerset name the value is the layerset itself.*

### Public Member Functions

- MadUtils ()
- QString openGraphicFile ()
- QString saveFile ()

### Static Public Member Functions

- static const QString userSettingsDirPath ()

  *userSettingsDirPath Find the place on the filesystem where user data is stored*
- static const QString userModelProfilesDirPath ()

  *uerModelProfilesDirPath Find the place on the filesystem where user defined model profiles are stored.*
- static const QString userModelParametersDirPath ()

  *userModelParametersDirPath Find the place on the filesystem where user defined model parameter profiles are stored.*
- static const QString getModelOutputDir ()

  *getModelOutputDir Get the place where model outputs are to be stored. By default this is in ∼/.macsurAdapter/model-Outputs But if modelOutputsDir is specified in QSettings, it will override the default.*
- static const QString userImagesDirPath ()

  *userImagesDirPath Find the place on the filesystem where user images are stored.*

- static MadUtils::ModelMap getAvailableModels ()

  *getAvailableModels Get a QMap of the avaliable layersets in the users layersets directory*
- static MadModel getModel (QString theGuid)

  *getModel Get a MadModel given its GUID. If no matching model is found, a blank one is returned.*
- static QStringList sortList (QStringList theList)

  *sortList Sort a string list into descending alphabetic order and return the result.*
- static QStringList uniqueList (QStringList theList)

  *uniqueList Remove any duplucate entries from a sorted list*
- static bool createTextFile (QString theFileName, QString theData)

  *createTextFile A helper method to easily write a file to disk.*
- static QString xmlEncode (QString theString)

  *xmlEncode A helper method to xml encode any special chars in a string (< > & etc) will become (< > & etc)*
- static QString xmlDecode (QString theString)

  *xmlDecode A helper method to xml deencode any special chars in a string (< > & etc) will become (< > & etc)*
- static QString getStandardCss ()

  *getStandardCss Get the standard style sheet for reports. Typically this will be used like this: QString myStyle = getStandardCss(); textBrowserFoo->document()->setDefaultStylesheet(myStyle);*
- static const QString userConversionTablesDirPath ()

### 6.22.1 Detailed Description

Definition at line 41 of file madutils.h.

### 6.22.2 Member Typedef Documentation

#### 6.22.2.1 typedef QMap<QString,MadModel> MadUtils::ModelMap

ModelMap (typedef) This typedef is used to refer to a collection of layersets. the key is the layerset name the value is the layerset itself.

Definition at line 101 of file madutils.h.

### 6.22.3 Constructor & Destructor Documentation

#### 6.22.3.1 MadUtils::MadUtils ( )

Definition at line 44 of file madutils.cpp.

```
45 {
46 }
```

### 6.22.4 Member Function Documentation

#### 6.22.4.1 bool MadUtils::createTextFile ( QString *theFileName*, QString *theData* ) [static]

createTextFile A helper method to easily write a file to disk.

**Parameters**

| | |
|---|---|
| *theFileName* | - the filename to be created or overwritten |
| *theData* | - the data that will be written into the file |

**Returns**

bool - false if the file could not be written

Definition at line 126 of file madutils.cpp.

```
{
    //create the txt file
  QFile myFile( theFileName );
  if ( myFile.open( QIODevice::WriteOnly ) )
  {
    QTextStream myQTextStream( &myFile );
    myQTextStream << theData;
  }
  else
  {
    return false;
  }
  myFile.close();
  return true ;
}
```

#### 6.22.4.2 MadUtils::ModelMap MadUtils::getAvailableModels ( ) [static]

getAvailableModels Get a QMap of the avaliable layersets in the users layersets directory

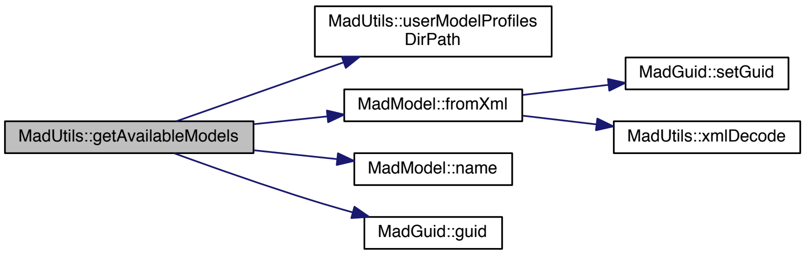
**Returns**

a QMap<QString,OmgLayerSet> where the QString key is the layerset name

Definition at line 93 of file madutils.cpp.

```
{
  MadUtils::ModelMap myMap;
  QDir myDirectory(userModelProfilesDirPath());
  myDirectory.setFilter(QDir::Dirs | QDir::Files | QDir::NoSymLinks );
  QFileInfoList myList = myDirectory.entryInfoList();
  for (unsigned int i = 0; i < static_cast<unsigned int>(myList.size()); ++i)
  {
    QFileInfo myFileInfo = myList.at(i);
      //Ignore directories
    if(myFileInfo.fileName() == "." ||myFileInfo.fileName() == ".." )
    {
      continue;
    }
      //if the filename ends in .xml try to load it into our layerSets listing
    if(myFileInfo.completeSuffix()=="xml")
    {
        //qDebug("Loading model: " + myList.at(i).absoluteFilePath().toLocal8Bit());
      MadModel myModel;
      myModel.fromXml(myFileInfo.absoluteFilePath());
      if (myModel.name().isEmpty())
      {
          //qDebug("Model name was empty!");
        continue;
      }
        //qDebug("Adding " + myModel.name());
      myMap[myModel.guid()]=myModel;
        //qDebug(myModel.toText().toLocal8Bit());
    }
  }
  return myMap;
}
```

Here is the call graph for this function:



#### 6.22.4.3 static MadModel MadUtils::getModel ( QString *theGuid* ) [static]

getModel Get a MadModel given its GUID. If no matching model is found, a blank one is returned.

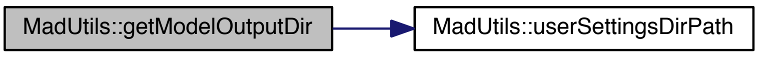
#### 6.22.4.4 const QString MadUtils::getModelOutputDir ( ) [static]

getModelOutputDir Get the place where model outputs are to be stored. By default this is in ∼/.macsur-Adapter/modelOutputs But if modelOutputsDir is specified in QSettings, it will override the default.

Definition at line 60 of file madutils.cpp.

```
{
  QString myPath = userSettingsDirPath()+QDir::separator()+"modelOutputs"+
   QDir::separator();
  QDir().mkpath(myPath);
  return myPath;
}
```

Here is the call graph for this function:



#### 6.22.4.5 QString MadUtils::getStandardCss ( ) [static]

getStandardCss Get the standard style sheet for reports. Typically this will be used like this: QString myStyle = getStandardCss(); textBrowserFoo->document()->setDefaultStylesheet(myStyle);

Definition at line 159 of file madutils.cpp.

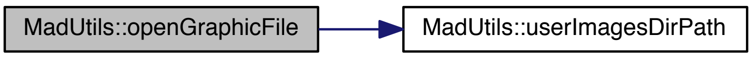
```
{
  QString myStyle = ".glossy{ background-color: qlineargradient(x1:0, y1:0, x2:0, y2:1, stop:0 #616161,
   stop: 0.5 #505050, stop: 0.6 #434343, stop:1 #656565); color: white; padding-left: 4px; border: 1px solid
   #6c6c6c; }";
  myStyle += "body {background: white;}";
  myStyle += "h1 {font-size : 22pt; color: #0063F7; }";
  myStyle += "h2 {font-size : 18pt; color: #0063F7; }";
  myStyle += "h3 {font-size : 14pt; color: #0063F7; }";
  myStyle += ".cellHeader {color:#466aa5; font-size : 12pt;}";
  myStyle += ".parameterHeader {font-weight: bold;}";
  myStyle += ".largeCell {color:#000000; font-size : 12pt;}";
  myStyle += ".table {"
                  "  border-width: 1px 1px 1px 1px;"
                  "  border-spacing: 2px;"
                  "  border-style: solid solid solid solid;"
                  "  border-color: black black black black;"
                  "  border-collapse: separate;"
                  "  background-color: white;"
                  "}";
  return myStyle;
}
```

#### 6.22.4.6 QString MadUtils::openGraphicFile ( )

Definition at line 180 of file madutils.cpp.

```
{
  QString myHomePath = QDir::homePath();
  QString myFileName = QFileDialog::getOpenFileName(0, "Choose an image", myHomePath, "Images (*.png *.xpm
   *.jpg)");
  QFileInfo fi(myFileName);
  QString myName = fi.fileName();
  QString myDestinationFilePathName = userImagesDirPath() + myName;
  QFile::copy(myFileName, myDestinationFilePathName);
  return myDestinationFilePathName;
}
```

Here is the call graph for this function:

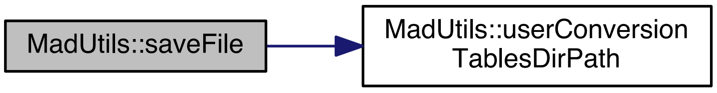


#### 6.22.4.7 QString MadUtils::saveFile ( )

Definition at line 191 of file madutils.cpp.

```
{
  QString myHomePath = QDir::homePath();
  QString myFileName = QFileDialog::getSaveFileName(0, "Choose a file name",
  userConversionTablesDirPath(), "*.csv");
  QFileInfo fi(myFileName);
  QString myName = fi.fileName();
  QString myDestinationFilePathName = userConversionTablesDirPath() + myName;
  return myDestinationFilePathName;
}
```

Here is the call graph for this function:



#### 6.22.4.8 static QStringList MadUtils::sortList ( QStringList *theList* ) [static]

sortList Sort a string list into descending alphabetic order and return the result.

**Parameters**

| | |
|---|---|
| *theList* | - the QStringList to be sorted |

**Returns**

QStringList - sorted in descending alphabetical order

#### 6.22.4.9 static QStringList MadUtils::uniqueList ( QStringList *theList* ) [static]

uniqueList Remove any duplucate entries from a sorted list

**Parameters**

| | |
|---|---|
| *theList* | - the QStringList to be sorted |

**Returns**

QStringList - a list with no sequential duplicates

#### 6.22.4.10 const QString MadUtils::userConversionTablesDirPath ( ) [static]

Find the place on the filesystem where user created conversion tables in csv format are stored

Typically this will be ∼/.macsurAdapter/conversionTables

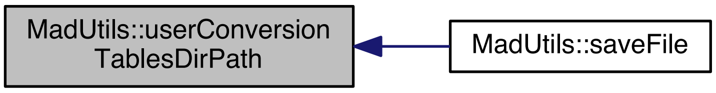
**Returns**

QString containing the relevant directory name

Definition at line 201 of file madutils.cpp.

```
{
   // always saved in the users home dir under .macsurAdapter/
  QString myPath = QDir::homePath() + QString("/.macsurAdapter/") +
    QDir::separator()+"conversionTables"+QDir::separator();
  QDir().mkpath(myPath);
  return myPath;
}
```

Here is the caller graph for this function:



#### 6.22.4.11 const QString MadUtils::userImagesDirPath ( ) [static]

userImagesDirPath Find the place on the filesystem where user images are stored.

Typically this will be ∼/.macsurAdapter/images

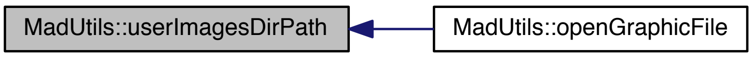
**Returns**

QString containing the relevant directory name

Definition at line 85 of file madutils.cpp.

```
{
  QString myPath = QDir::homePath() + QString("/.macsurAdapter") +
    QDir::separator()+"images"+QDir::separator();
  QDir().mkpath(myPath);
  return myPath;
}
```

Here is the caller graph for this function:



#### 6.22.4.12 const QString MadUtils::userModelParametersDirPath ( ) [static]

userModelParametersDirPath Find the place on the filesystem where user defined model parameter profiles are stored.

Typically this will be ∼/.macsurAdapter/animalParameters

**Returns**

QString containing the relevant directory name

Definition at line 76 of file madutils.cpp.

```
{
    //alg profiles are always saved in the users home dir under .macsurAdapter/
  QString myPath = QDir::homePath() + QString("/.macsurAdapter/") +
    QDir::separator() + "modelParameterProfiles" + QDir::separator();
  QDir().mkpath(myPath);
  return myPath;
}
```

#### 6.22.4.13 const QString MadUtils::userModelProfilesDirPath ( ) [static]

uerModelProfilesDirPath Find the place on the filesystem where user defined model profiles are stored.

Typically this will be ∼/.macsurAdapter/modelProfiles

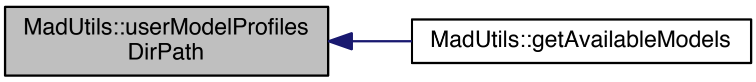
**Returns**

QString containing the relevant directory name

Definition at line 67 of file madutils.cpp.

```
{
    //alg profiles are always saved in the users home dir under .macsurAdapter
    QString myPath = QDir::homePath() + QString("/.macsurAdapter/") +
      QDir::separator()+"animalProfiles"+QDir::separator();
    QDir().mkpath(myPath);
    return myPath;
}
```

Here is the caller graph for this function:


MadUtils::userModelProfiles DirPath
MadUtils::getAvailableModels

#### 6.22.4.14 const QString MadUtils::userSettingsDirPath ( ) [static]

userSettingsDirPath Find the place on the filesystem where user data is stored

Typically, this will be ∼/.macsurAdapter

**Returns**

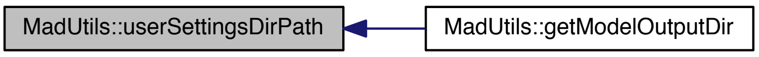
QString containing the relevant directory name

Returns the path to the settings directory in user's home dir

Definition at line 51 of file madutils.cpp.

```
{
  QSettings mySettings;
  QString myPath=
      mySettings.value("dataDirs/dataDir", QDir::homePath() + QString("/.macsurAdapter/") ).toString();
  //  Make sure the users settings dir actually exists
  QDir().mkpath(myPath);
  return myPath;
}
```

Here is the caller graph for this function:



#### 6.22.4.15 QString MadUtils::xmlDecode ( QString *theString* ) [static]

xmlDecode A helper method to xml deencode any special chars in a string (< > & etc) will become (< > & etc)

**Parameters**

| | |
|---|---|
| *QString* | - the string to be properly decoded |

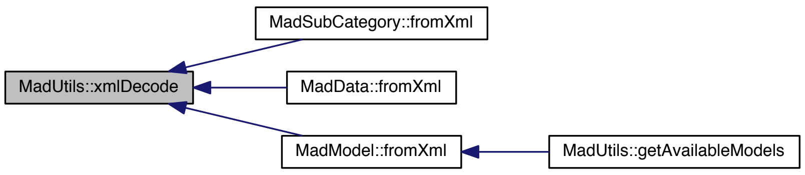
**Returns**

A QString with the encoded chars properly decoded

Definition at line 151 of file madutils.cpp.

```
{
  theString = theString.replace("&lt;","<");
  theString = theString.replace("&gt;",">");
  theString = theString.replace("&amp;","&");
  return theString;
}
```

Here is the caller graph for this function:



#### 6.22.4.16 QString MadUtils::xmlEncode ( QString *theString* ) [static]

xmlEncode A helper method to xml encode any special chars in a string (< > & etc) will become (< > & etc)

**Parameters**

| | |
|---|---|
| *QString* | - the string to be properly encoded |

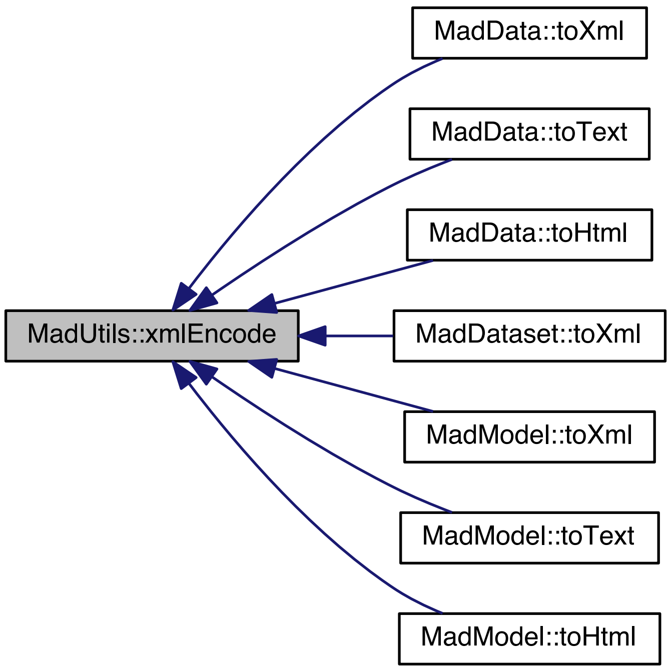
**Returns**

A QString with the special chars properly encoded

Definition at line 143 of file madutils.cpp.

```
{
  theString = theString.replace("<","&lt;");
  theString = theString.replace(">","&gt;");
  theString = theString.replace("&","&amp;");
  return theString;
}
```

Here is the caller graph for this function:

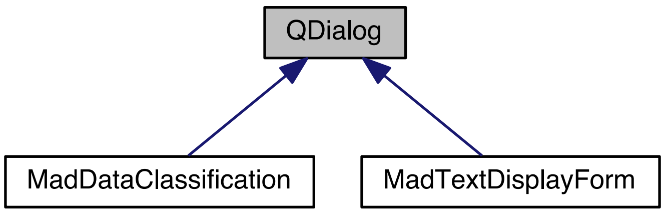


The documentation for this class was generated from the following files:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madutils.h
- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madutils.cpp

## 6.23 QDialog Class Reference

Inheritance diagram for QDialog:

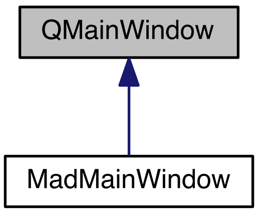


The documentation for this class was generated from the following file:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/madtextdisplayform.h

## 6.24 QMainWindow Class Reference

Inheritance diagram for QMainWindow:



The documentation for this class was generated from the following file:

- /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/madmainwindow.h

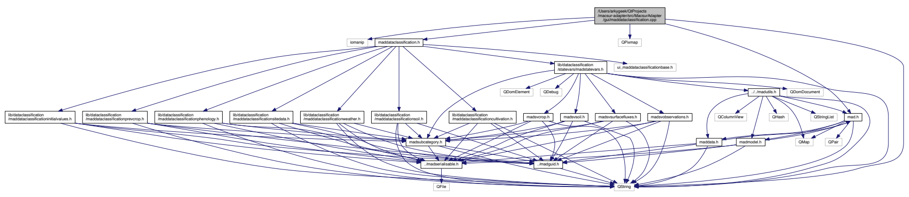
# Chapter 7

# File Documentation

## 7.1 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/maddataclassification.cpp File Reference

```
#include <iomanip>
#include <QString>
#include <QPixmap>
#include "maddataclassification.h"
#include "lib/mad.h"
```

Include dependency graph for maddataclassification.cpp:



### Functions

- QString makeString ()

### 7.1.1 Function Documentation

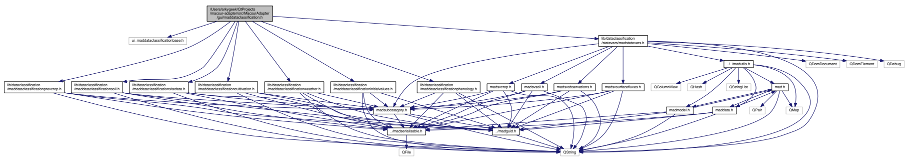
#### 7.1.1.1 QString makeString ( )

## 7.2 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/maddataclassification.h File Reference

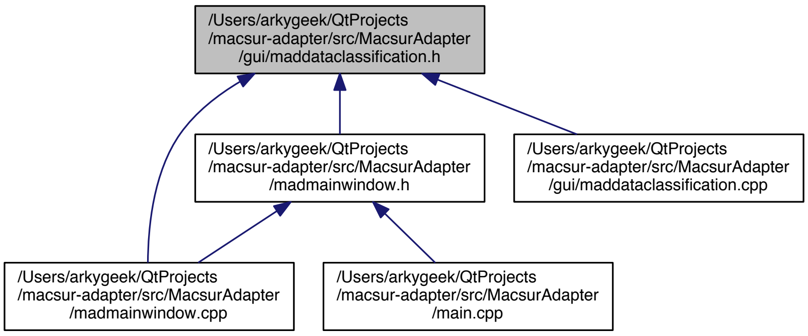
```
#include "ui_maddataclassificationbase.h"
```

```
#include "lib/dataclassification/maddataclassificationcultivation.h"
#include "lib/dataclassification/maddataclassificationinitialvalues.h"
#include "lib/dataclassification/maddataclassificationphenology.h"
#include "lib/dataclassification/maddataclassificationprevcrop.h"
#include "lib/dataclassification/maddataclassificationsitedata.h"
#include "lib/dataclassification/maddataclassificationsoil.h"
#include "lib/dataclassification/maddataclassificationweather.h"
#include "lib/dataclassification/statevars/madstatevars.h"
```

Include dependency graph for maddataclassification.h:



This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /main.cpp

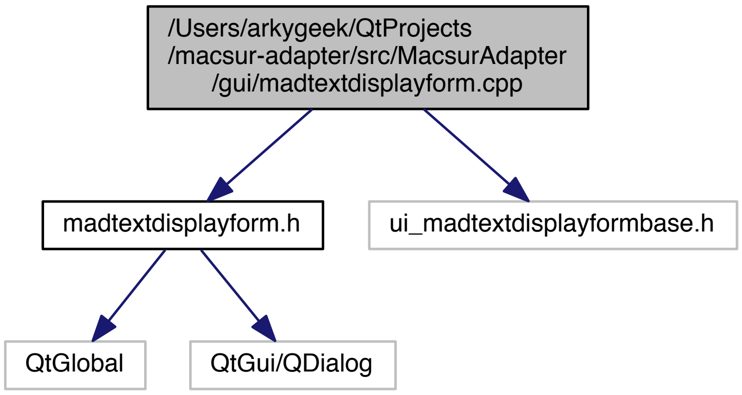
### Classes

- class MadDataClassification

## 7.3 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/madtextdisplayform.cpp File Reference

```
#include "madtextdisplayform.h"
#include "ui_madtextdisplayformbase.h"
```

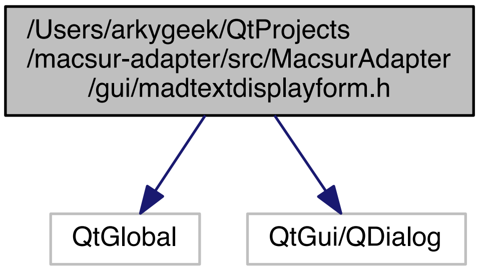
Include dependency graph for madtextdisplayform.cpp:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/gui/madtextdisplayform.cpp
madtextdisplayform.h
ui_madtextdisplayformbase.h
QtGlobal
QtGui/QDialog

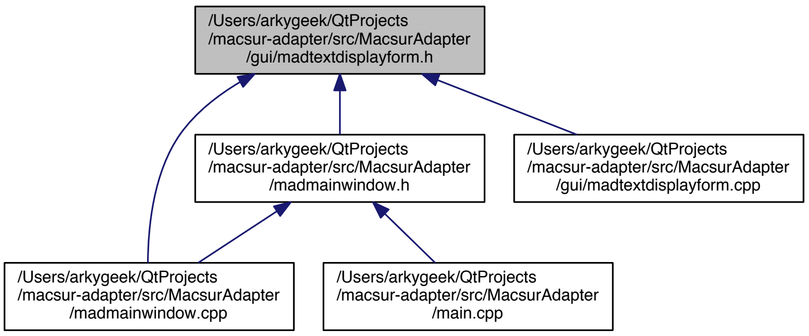
## 7.4 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/madtextdisplayform.h File Reference

```
#include <QtGlobal>
#include <QtGui/QDialog>
```

Include dependency graph for madtextdisplayform.h:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/gui/madtextdisplayform.h
QtGlobal
QtGui/QDialog

This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/madtextdisplayform.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/madtextdisplayform.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /main.cpp

### Classes

- class MadTextDisplayForm

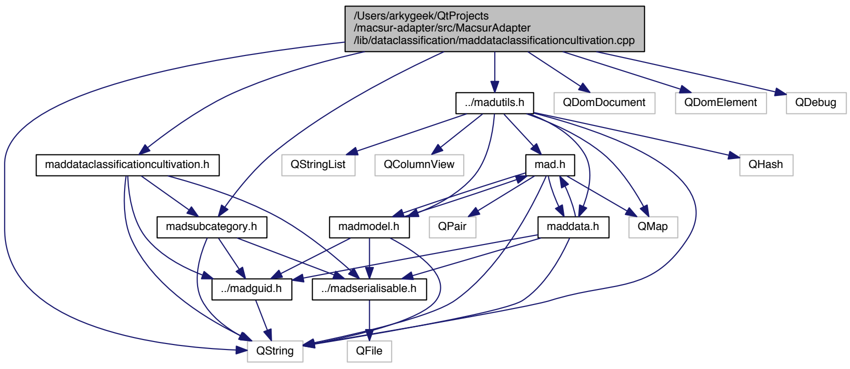
### Namespaces

- namespace Ui

## 7.5 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclass File Reference

```
#include "maddataclassificationcultivation.h"
#include "madsubcategory.h"
#include "../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

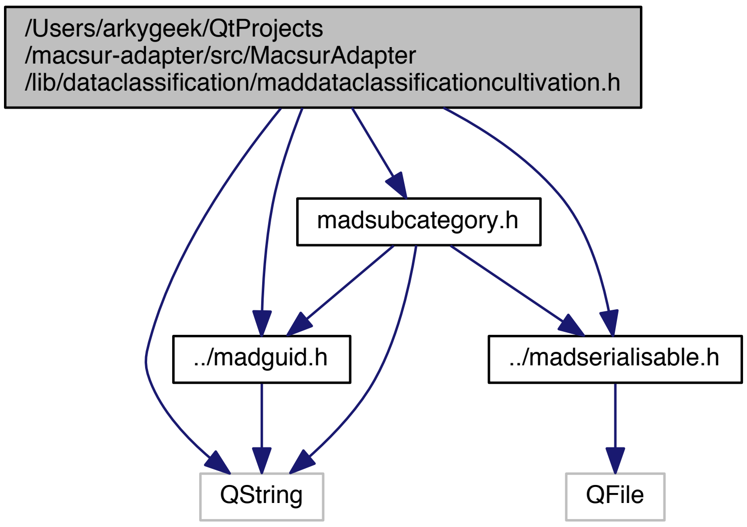
Include dependency graph for maddataclassificationcultivation.cpp:



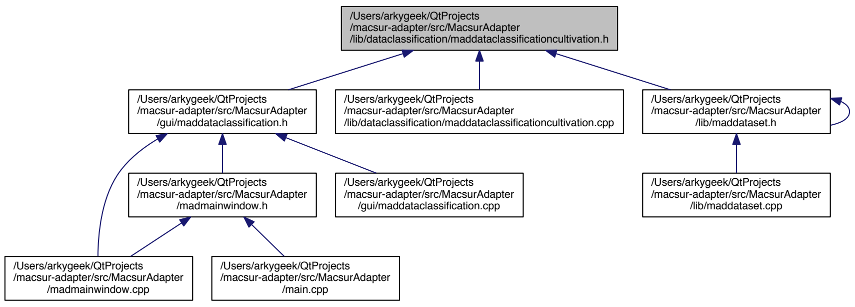
## 7.6 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclass File Reference

```
#include "madsubcategory.h"
#include "../madguid.h"
#include "../madserialisable.h"
#include <QString>
```

Include dependency graph for maddataclassificationcultivation.h:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/maddataclassificationcultivation.h
madsubcategory.h
../madguid.h
../madserialisable.h
QString
QFile

This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/maddataclassificationcultivation.h
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/gui/maddataclassification.h
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/maddataclassificationcultivation.cpp
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/maddataset.h
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/madmainwindow.h
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/gui/maddataclassification.cpp
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/maddataset.cpp
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/madmainwindow.cpp
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/main.cpp

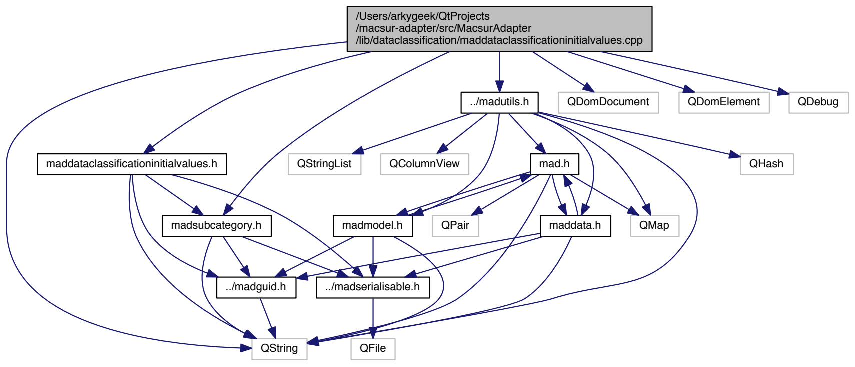
### Classes

- class MadDataClassificationCultivation

## 7.7 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclass File Reference

```
#include "maddataclassificationinitialvalues.h"
#include "madsubcategory.h"
#include "../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

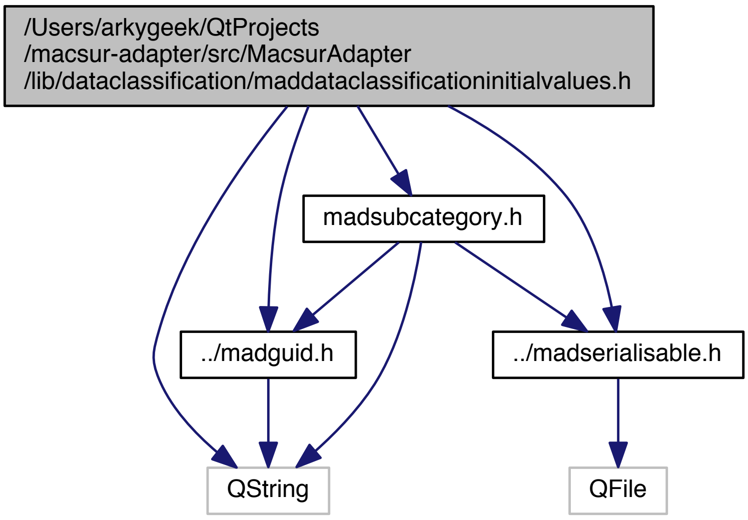
Include dependency graph for maddataclassificationinitialvalues.cpp:



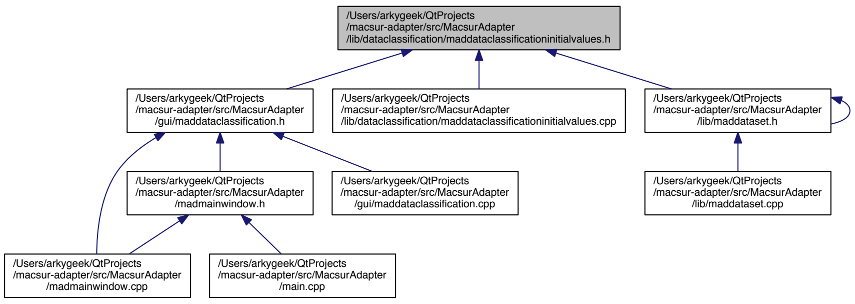
## 7.8 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclass File Reference

```
#include "madsubcategory.h"
#include "../madguid.h"
#include "../madserialisable.h"
#include <QString>
```

Include dependency graph for maddataclassificationinitialvalues.h:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/maddataclassificationinitialvalues.h
madsubcategory.h
../madguid.h
../madserialisable.h
QString
QFile

This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/maddataclassificationinitialvalues.h
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/gui/maddataclassification.h
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/maddataclassificationinitialvalues.cpp
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/maddataset.h
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/madmainwindow.h
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/gui/maddataclassification.cpp
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/maddataset.cpp
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/madmainwindow.cpp
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/main.cpp

## Classes

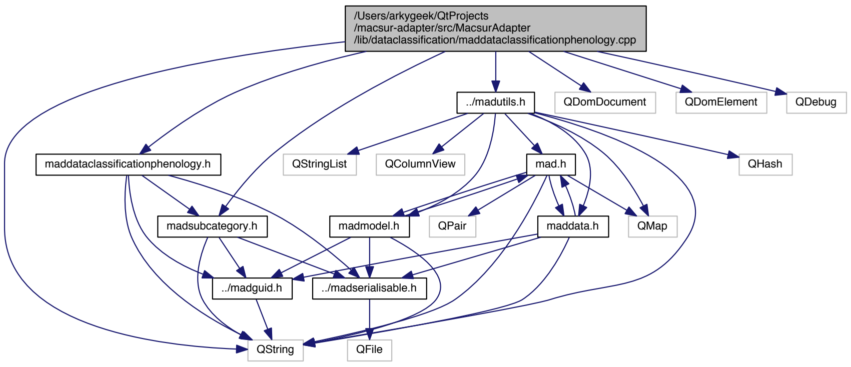
- class MadDataClassificationInitialValues

## 7.9 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclass File Reference

```
#include "maddataclassificationphenology.h"
```

```
#include "madsubcategory.h"
#include "../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

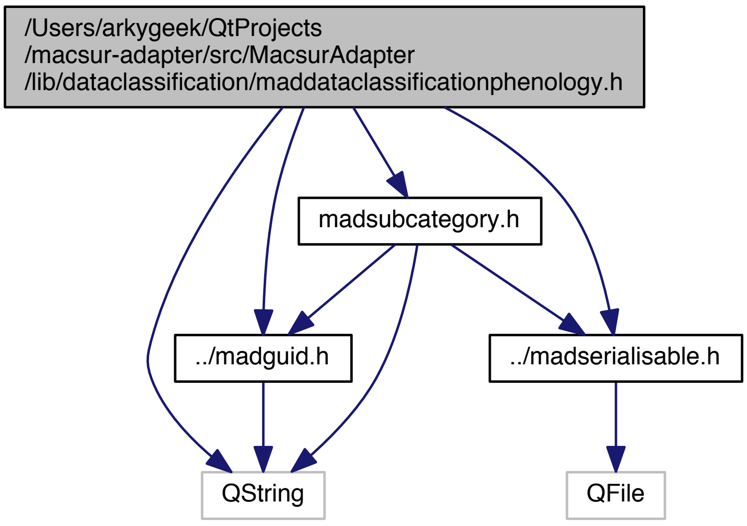
Include dependency graph for maddataclassificationphenology.cpp:



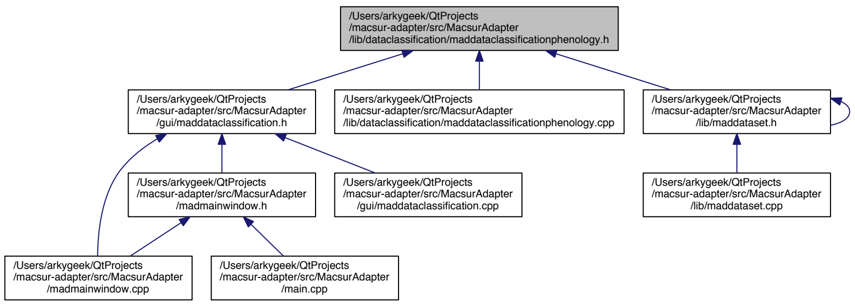
## 7.10 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclas File Reference

```
#include "madsubcategory.h"
#include "../madguid.h"
#include "../madserialisable.h"
#include <QString>
```

Include dependency graph for maddataclassificationphenology.h:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/maddataclassificationphenology.h
madsubcategory.h
../madguid.h
../madserialisable.h
QString
QFile

This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/maddataclassificationphenology.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/maddataclassificationphenology.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /main.cpp

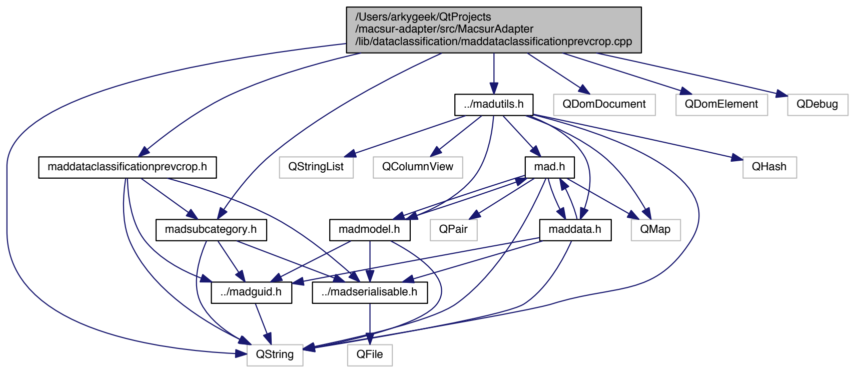
## Classes

- class MadDataClassificationPhenology

## 7.11 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclas File Reference

```
#include "maddataclassificationprevcrop.h"
#include "madsubcategory.h"
#include "../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

Include dependency graph for maddataclassificationprevcrop.cpp:

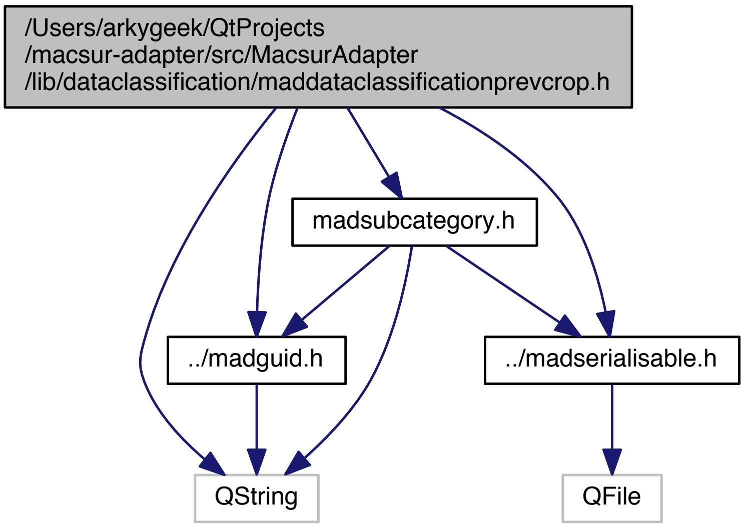


## 7.12 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclas File Reference

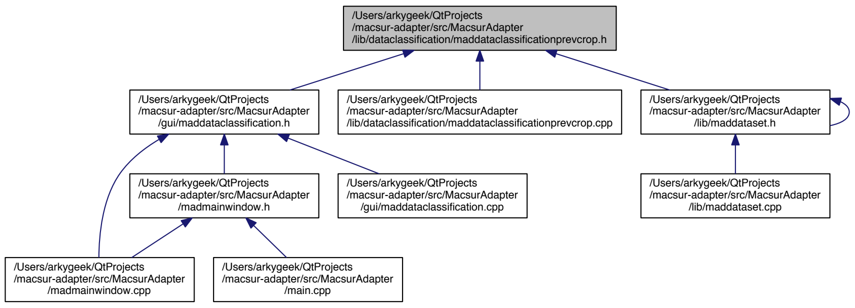
```
#include "madsubcategory.h"
```

```
#include "../madguid.h"
#include "../madserialisable.h"
#include <QString>
```

Include dependency graph for maddataclassificationprevcrop.h:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/maddataclassificationprevcrop.h
madsubcategory.h
../madguid.h
../madserialisable.h
QString
QFile

This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/maddataclassificationprevcrop.h
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/gui/maddataclassification.h
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/maddataclassificationprevcrop.cpp
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/maddataset.h
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/madmainwindow.h
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/gui/maddataclassification.cpp
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/maddataset.cpp
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/madmainwindow.cpp
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/main.cpp

### Classes

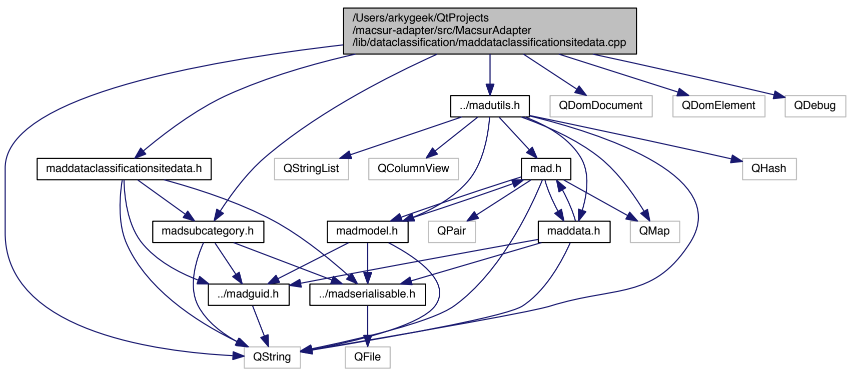
- class MadDataClassificationPrevCrop

## 7.13 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclas File Reference

```
#include "maddataclassificationsitedata.h"
```

```
#include "madsubcategory.h"
#include "../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

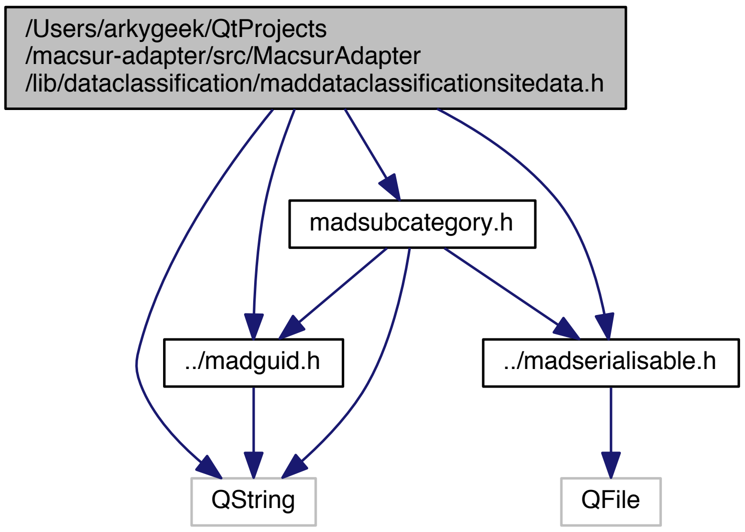
Include dependency graph for maddataclassificationsitedata.cpp:



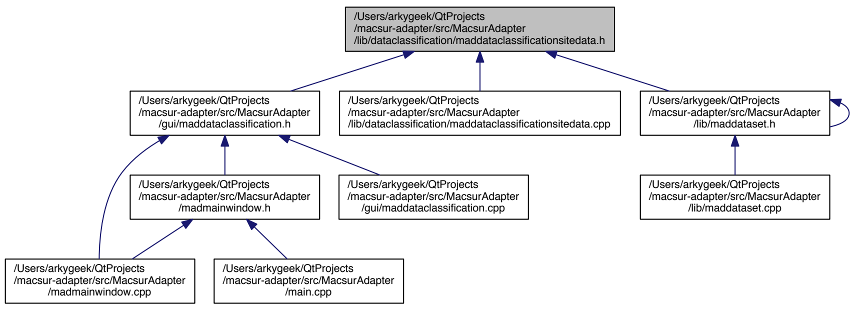
## 7.14 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclas File Reference

```
#include "madsubcategory.h"
#include "../madguid.h"
#include "../madserialisable.h"
#include <QString>
```

Include dependency graph for maddataclassificationsitedata.h:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/maddataclassificationsitedata.h
madsubcategory.h
../madguid.h
../madserialisable.h
QString
QFile

This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/maddataclassificationsitedata.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/maddataclassificationsitedata.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /main.cpp

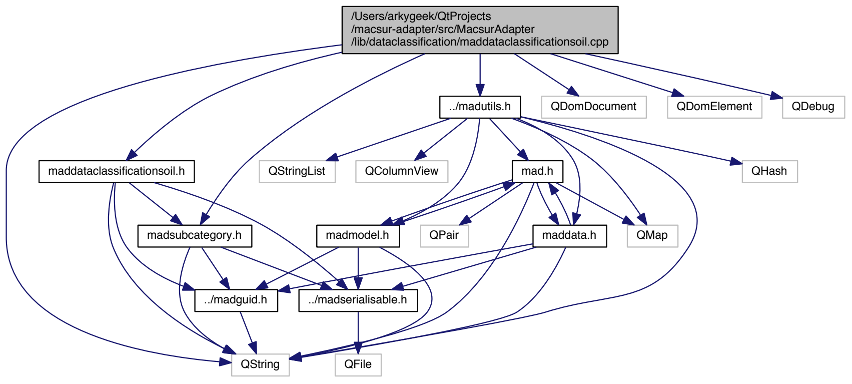
### Classes

- class MadDataClassificationSiteData

## 7.15 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclas File Reference

```
#include "maddataclassificationsoil.h"
#include "madsubcategory.h"
#include "../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

Include dependency graph for maddataclassificationsoil.cpp:

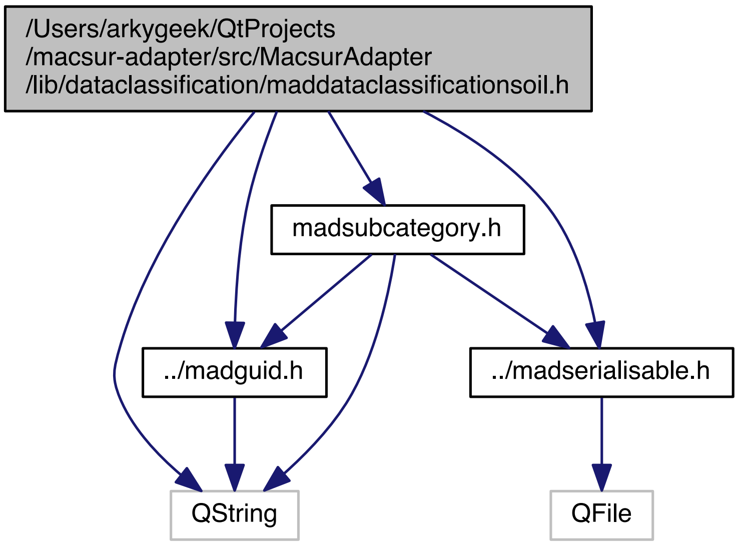


## 7.16 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclas File Reference

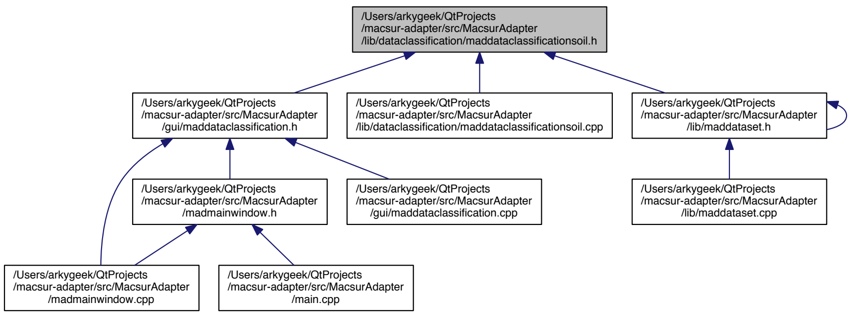
```
#include "madsubcategory.h"
```

```
#include "../madguid.h"
#include "../madserialisable.h"
#include <QString>
```

Include dependency graph for maddataclassificationsoil.h:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/maddataclassificationsoil.h
madsubcategory.h
../madguid.h
../madserialisable.h
QString
QFile

This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/maddataclassificationsoil.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/maddataclassificationsoil.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /main.cpp

### Classes

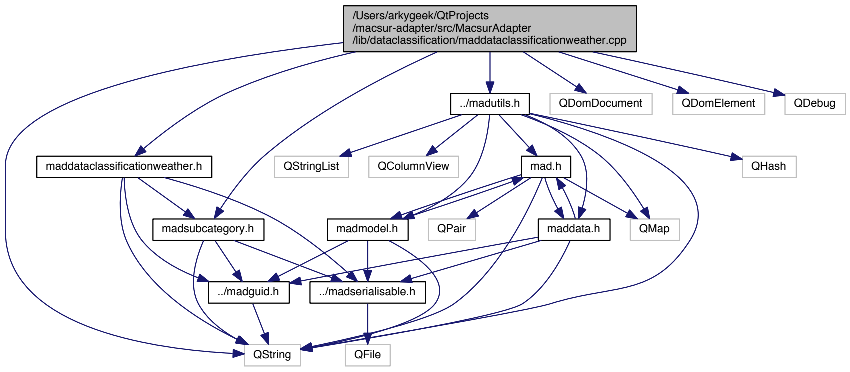
- class MadDataClassificationSoil

## 7.17 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclas File Reference

```
#include "maddataclassificationweather.h"
```

```
#include "madsubcategory.h"
#include "../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

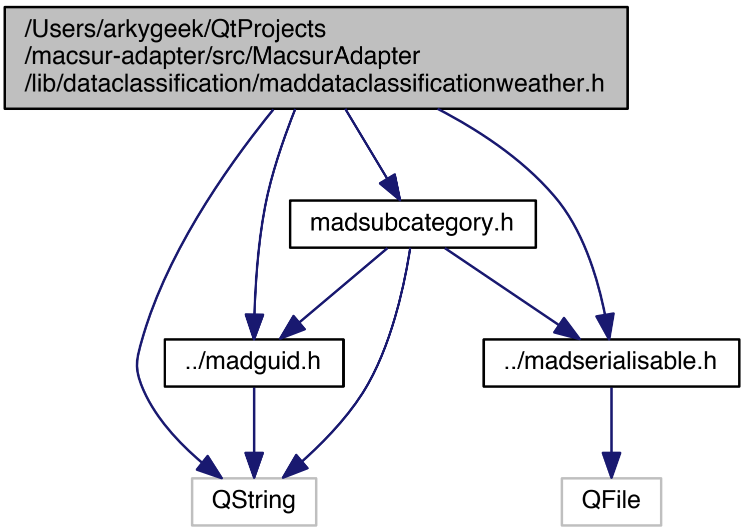
Include dependency graph for maddataclassificationweather.cpp:



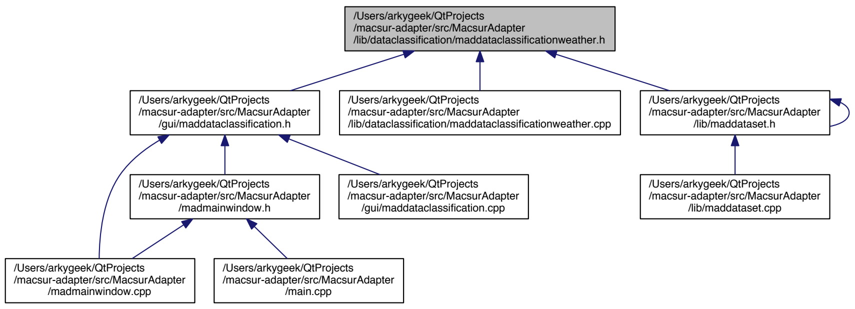
## 7.18 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclas File Reference

```
#include "madsubcategory.h"
#include "../madguid.h"
#include "../madserialisable.h"
#include <QString>
```

Include dependency graph for maddataclassificationweather.h:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/maddataclassificationweather.h
madsubcategory.h
../madguid.h
../madserialisable.h
QString
QFile

This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationweather.h
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/maddataclassification.h
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/maddataclassificationweather.cpp
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddataset.h
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/madmainwindow.h
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/maddataclassification.cpp
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddataset.cpp
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/madmainwindow.cpp
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/main.cpp

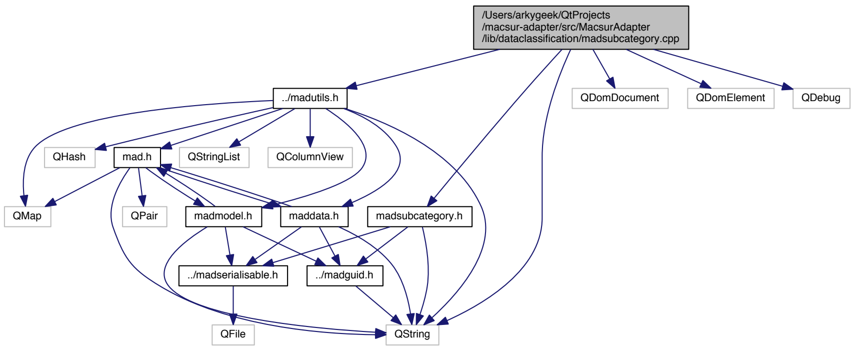
### Classes

- class MadDataClassificationWeather

## 7.19 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/madsubcateg File Reference

```
#include "madsubcategory.h"
#include "../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

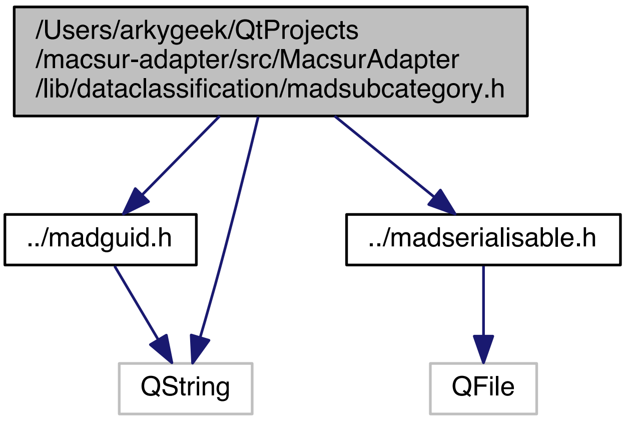
Include dependency graph for madsubcategory.cpp:



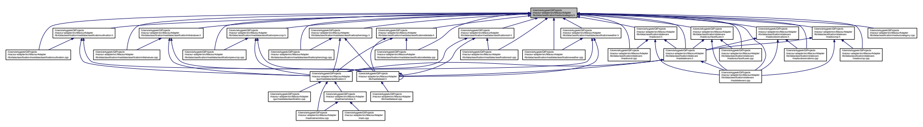
## 7.20 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/madsubcateg File Reference

```
#include "../madguid.h"
#include "../madserialisable.h"
#include <QString>
```

Include dependency graph for madsubcategory.h:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/madsubcategory.h
../madguid.h
../madserialisable.h
QString
QFile

This graph shows which files directly or indirectly include this file:



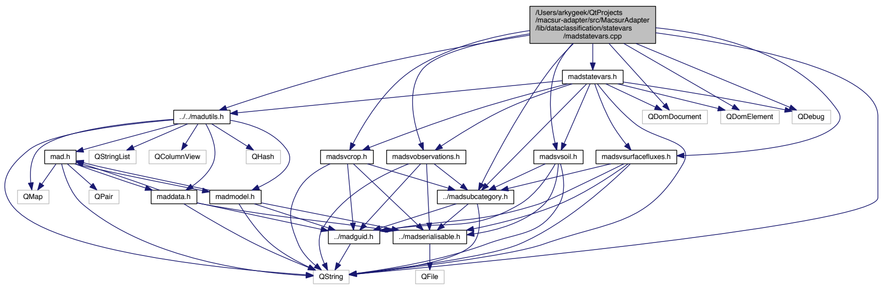
## Classes

- class MadSubCategory

## 7.21 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/ma File Reference

```
#include "madstatevars.h"
#include "madsvcrop.h"
#include "madsvsoil.h"
#include "madsvsurfacefluxes.h"
#include "madsvobservations.h"
#include "../madsubcategory.h"
#include "../../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

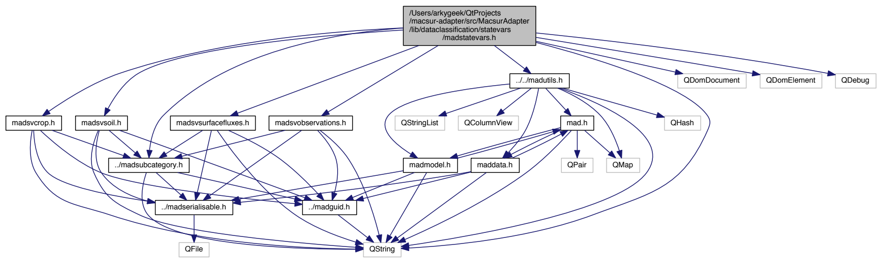
Include dependency graph for madstatevars.cpp:



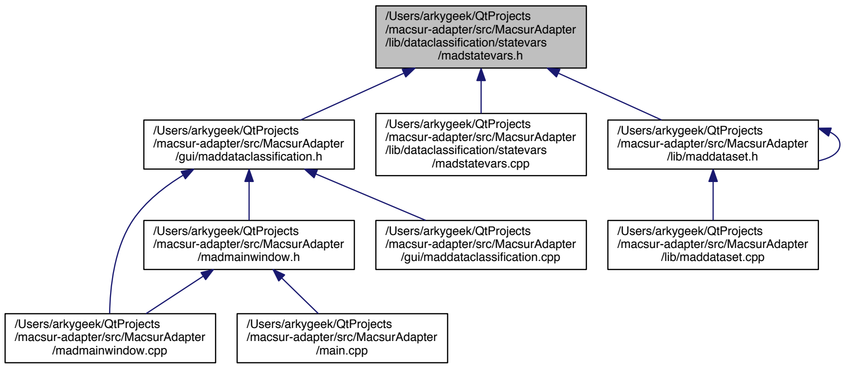
## 7.22 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/ma File Reference

```
#include "madsvcrop.h"
#include "madsvsoil.h"
#include "madsvsurfacefluxes.h"
#include "madsvobservations.h"
#include "../madsubcategory.h"
#include "../../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

Include dependency graph for madstatevars.h:



This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madstatevars.h
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/maddataclassification.h
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madstatevars.cpp
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddataset.h
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/madmainwindow.h
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/maddataclassification.cpp
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddataset.cpp
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/madmainwindow.cpp
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/main.cpp

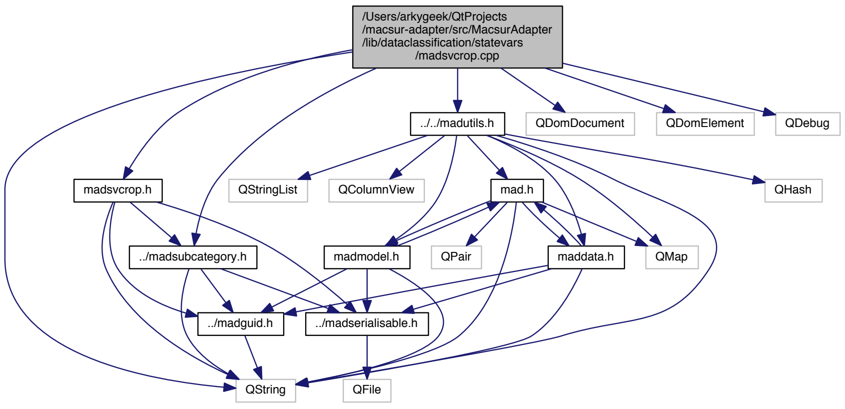
## Classes

- class MadStateVars

## 7.23 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/ma File Reference

```
#include "madsvcrop.h"
#include "../madsubcategory.h"
#include "../../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

Include dependency graph for madsvcrop.cpp:

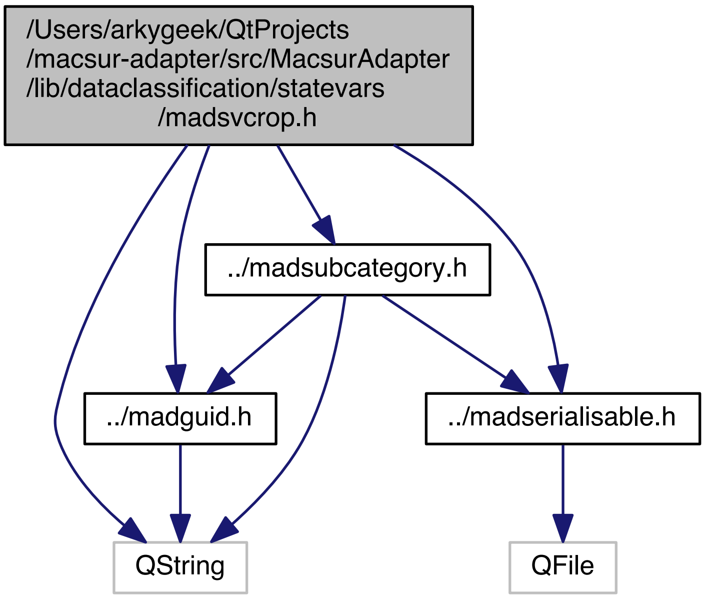


## 7.24 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/ma File Reference

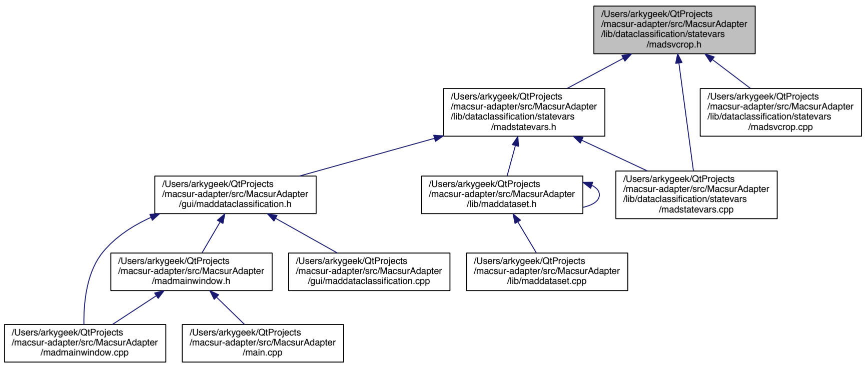
The MadStateVars class. This contains 4 sub categories.

```
#include "../madsubcategory.h"
#include "../../madguid.h"
#include "../../madserialisable.h"
#include <QString>
```

Include dependency graph for madsvcrop.h:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madsvcrop.h
../madsubcategory.h
../madguid.h
../madserialisable.h
QString
QFile

This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madsvcrop.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madstatevars.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madsvcrop.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madstatevars.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /main.cpp

### Classes

- class MadSVCrop

### 7.24.1 Detailed Description

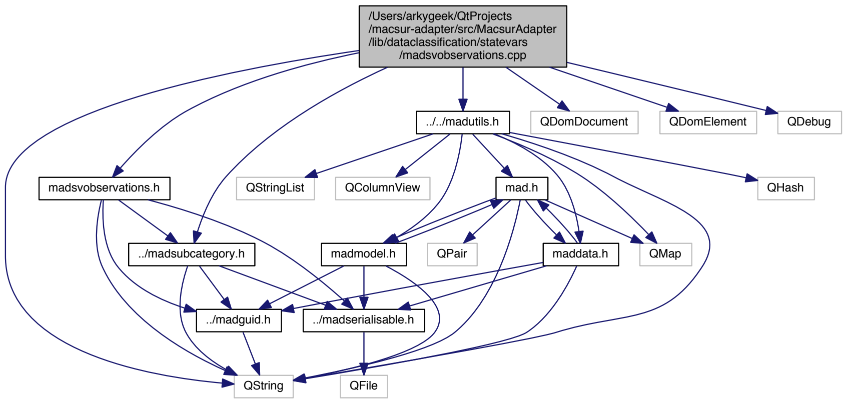
The MadStateVars class. This contains 4 sub categories.

Definition in file madsvcrop.h.

## 7.25 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/ma File Reference

```
#include "madsvobservations.h"
#include "../madsubcategory.h"
#include "../../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

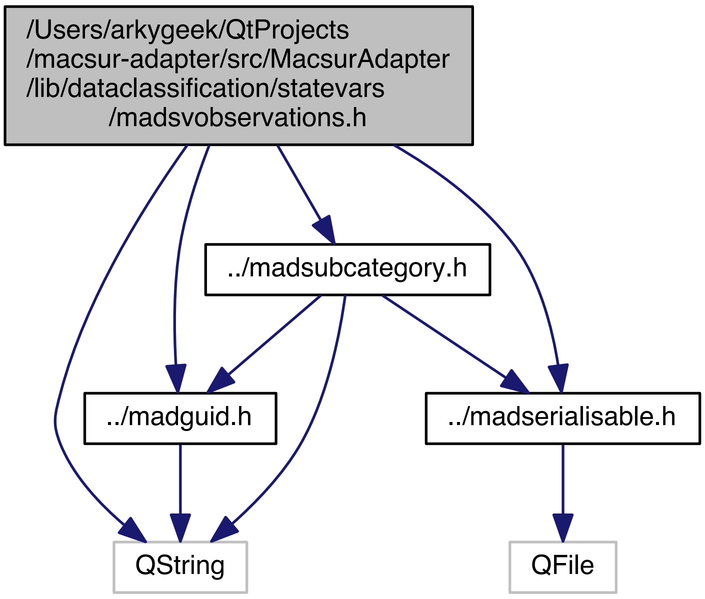
Include dependency graph for madsvobservations.cpp:



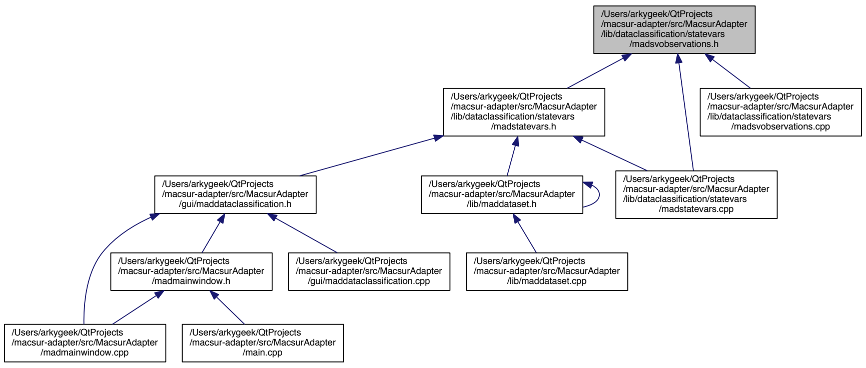
## 7.26 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/ma File Reference

```
#include "../madsubcategory.h"
#include "../../madguid.h"
#include "../../madserialisable.h"
#include <QString>
```

Include dependency graph for madsvobservations.h:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/statevars
/madsvobservations.h
../madsubcategory.h
../madguid.h
../madserialisable.h
QString
QFile

This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madsvobservations.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madstatevars.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madsvobservations.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madstatevars.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /main.cpp

## Classes

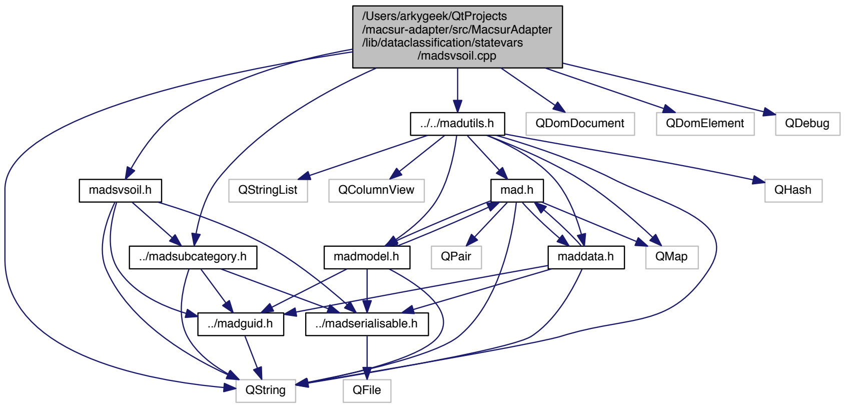
- class MadSVObservations

## 7.27 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/ma File Reference

```
#include "madsvsoil.h"
```

```
#include "../madsubcategory.h"
#include "../../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

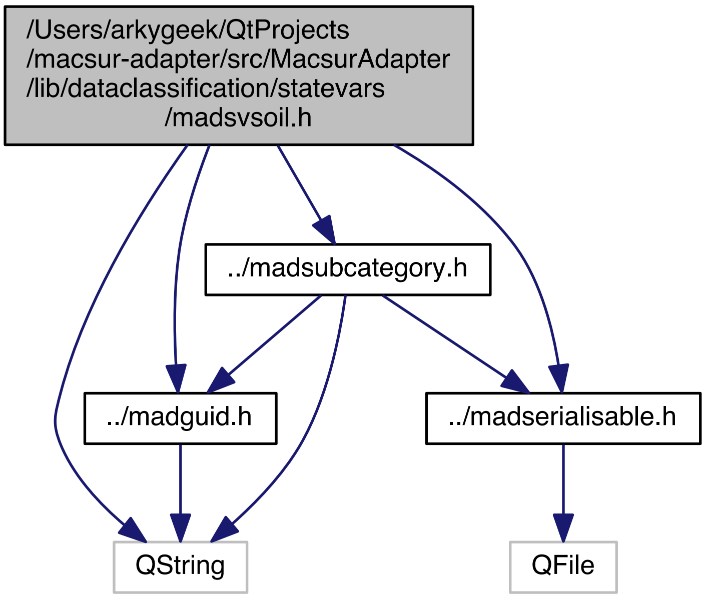
Include dependency graph for madsvsoil.cpp:



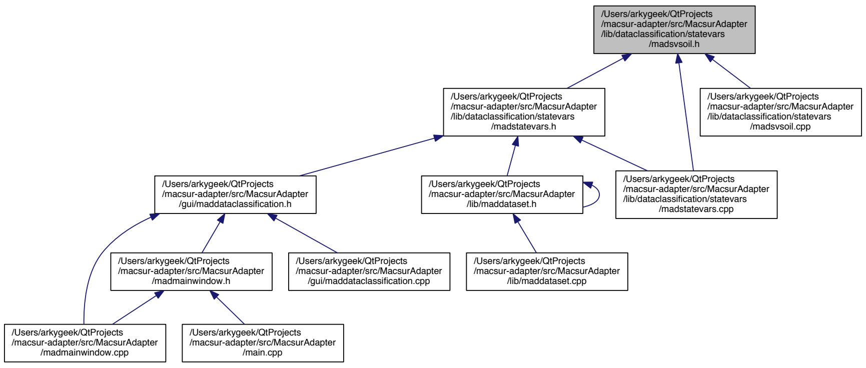
## 7.28 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/ma File Reference

```
#include "../madsubcategory.h"
#include "../../madguid.h"
#include "../../madserialisable.h"
#include <QString>
```

Include dependency graph for madsvsoil.h:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/lib/dataclassification/statevars
/madsvsoil.h
../madsubcategory.h
../madguid.h
../madserialisable.h
QString
QFile

This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madsvsoil.h
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madstatevars.h
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madsvsoil.cpp
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/maddataclassification.h
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddataset.h
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/madstatevars.cpp
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/madmainwindow.h
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/gui/maddataclassification.cpp
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddataset.cpp
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/madmainwindow.cpp
/Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/main.cpp

## Classes

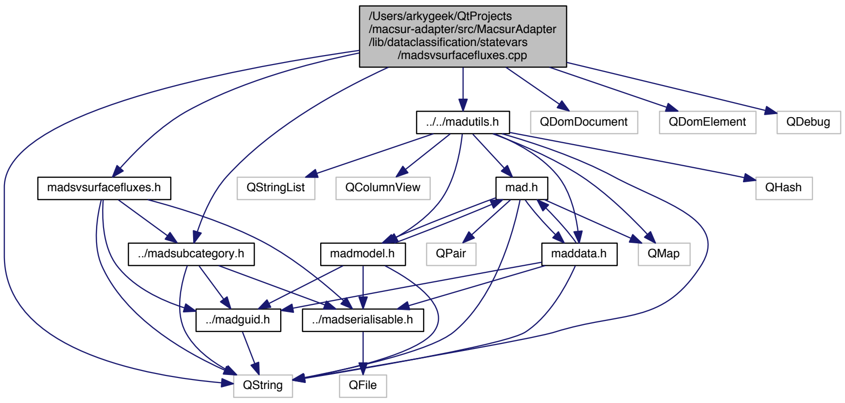
- class MadSVSoil

## 7.29 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/ma File Reference

```
#include "madsvsurfacefluxes.h"
```

```
#include "../madsubcategory.h"
#include "../../madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

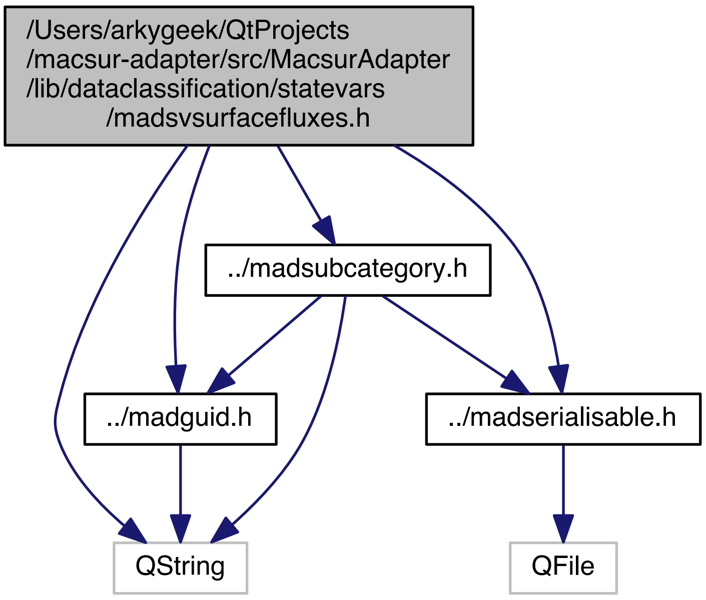
Include dependency graph for madsvsurfacefluxes.cpp:



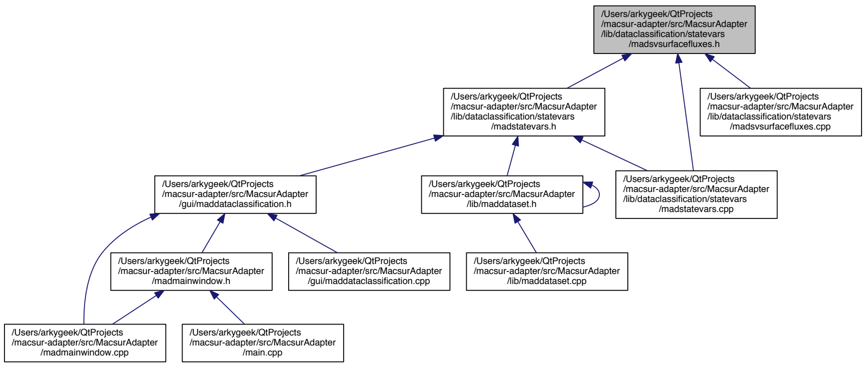
## 7.30 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/dataclassification/statevars/ma File Reference

```
#include "../madsubcategory.h"
#include "../../madguid.h"
#include "../../madserialisable.h"
#include <QString>
```

Include dependency graph for madsvsurfacefluxes.h:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madsvsurfacefluxes.h
../madsubcategory.h
../madguid.h
../madserialisable.h
QString
QFile

This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madsvsurfacefluxes.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madstatevars.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madsvsurfacefluxes.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/dataclassification/statevars /madstatevars.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /gui/maddataclassification.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /main.cpp

## Classes

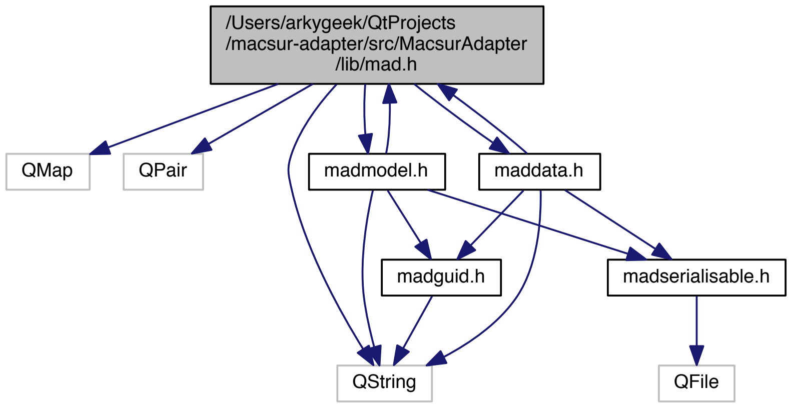
- class MadSVSurfaceFluxes

  *The MadSVSurfaceFluxes class.*

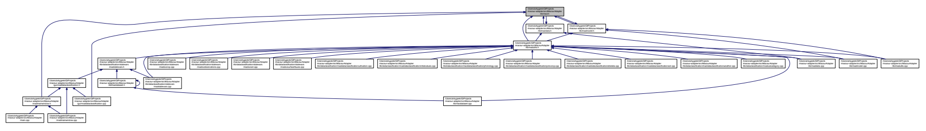
## 7.31 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/mad.h File Reference

```
#include <QMap>
#include <QPair>
#include <QString>
#include "madmodel.h"
#include "maddata.h"
```

Include dependency graph for mad.h:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/mad.h
QMap
QPair
madmodel.h
maddata.h
madguid.h
madserialisable.h
QString
QFile

This graph shows which files directly or indirectly include this file:



### Typedefs

- typedef QMap< QString, QPair < bool, QString > > MadTripleMap

  *MadTripleMap.*
- typedef QPair< QPair< QString, QString >, QPair< QString, QString > > MadModelInfo

  *MadModelInfo.*

### Enumerations

- enum ModelTheme { CropM, LiveM, TradeM }

  *MadModelMap.*
- enum Scale { Farm, Locality, Regional, National, International, Global }

*The Scale enum.*
- enum Nuts { Nuts1, Nuts2, Nuts3 }

  *The Nuts enum.*
- enum AreaUnits {
  Dunum, Hectare, Acre, SquareKm,
  SquareMile }

  *The AreaUnits enum.*
- enum FileType { CSV, TAB, OtherDelimited, Binary }

  *The FileType enum.*
- enum EnergyType { KCalories, TDN }

  *The EnergyType enum.*
- enum DataClass { Platinum, Gold, Silver, Bronze }

  *The DataClass enum.*

## 7.31.1 Typedef Documentation

### 7.31.1.1 typedef QPair<QPair<QString,QString>, QPair<QString,QString> > **MadModelInfo**

MadModelInfo.

Definition at line 51 of file mad.h.

### 7.31.1.2 typedef QMap<QString,QPair<bool,QString> > **MadTripleMap**

MadTripleMap.

Definition at line 47 of file mad.h.

## 7.31.2 Enumeration Type Documentation

### 7.31.2.1 enum **AreaUnits**

The AreaUnits enum.

**Enumerator**

***Dunum***

***Hectare***

***Acre***

***SquareKm***

***SquareMile***

Definition at line 72 of file mad.h.

```
72 {Dunum, Hectare, Acre, SquareKm, SquareMile};
```

### 7.31.2.2 enum **DataClass**

The DataClass enum.

**Enumerator**

***Platinum***

***Gold***

***Silver***

***Bronze***

Definition at line 84 of file mad.h.

```
84 {Platinum, Gold, Silver, Bronze};
```

#### 7.31.2.3 enum EnergyType

The EnergyType enum.

**Enumerator**

***KCalories***

***TDN***

Definition at line 80 of file mad.h.

```
80 {KCalories, TDN};
```

#### 7.31.2.4 enum FileType

The FileType enum.

**Enumerator**

***CSV***

***TAB***

***OtherDelimited***

***Binary***

Definition at line 76 of file mad.h.

```
76 {CSV, TAB, OtherDelimited, Binary};
```

#### 7.31.2.5 enum ModelTheme

MadModelMap.

The ModelTheme enum

**Enumerator**

***CropM***

***LiveM***

***TradeM***

Definition at line 60 of file mad.h.

```
60 {CropM, LiveM, TradeM};
```

#### 7.31.2.6 enum Nuts

The Nuts enum.

**Enumerator**

***Nuts1***
***Nuts2***
***Nuts3***

Definition at line 68 of file mad.h.

```
68 {Nuts1, Nuts2, Nuts3};
```

#### 7.31.2.7 enum Scale

The Scale enum.

**Enumerator**

***Farm***
***Locality***
***Regional***
***National***
***International***
***Global***

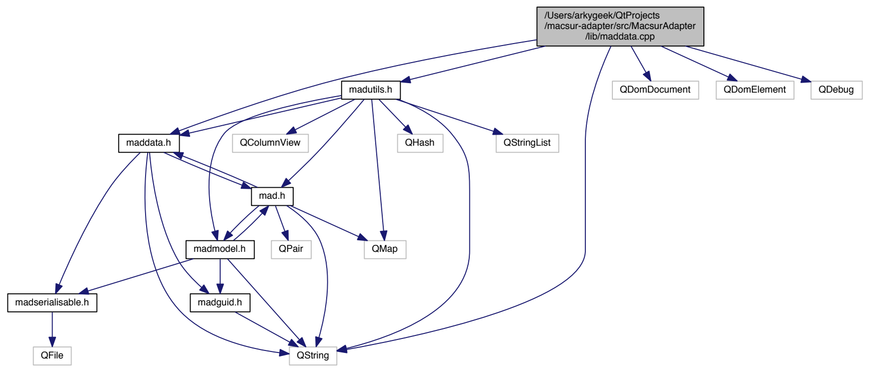
Definition at line 64 of file mad.h.

```
64 {Farm, Locality, Regional, National, International,
      Global};
```

## 7.32 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddata.cpp File Reference

```
#include "maddata.h"
#include "madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

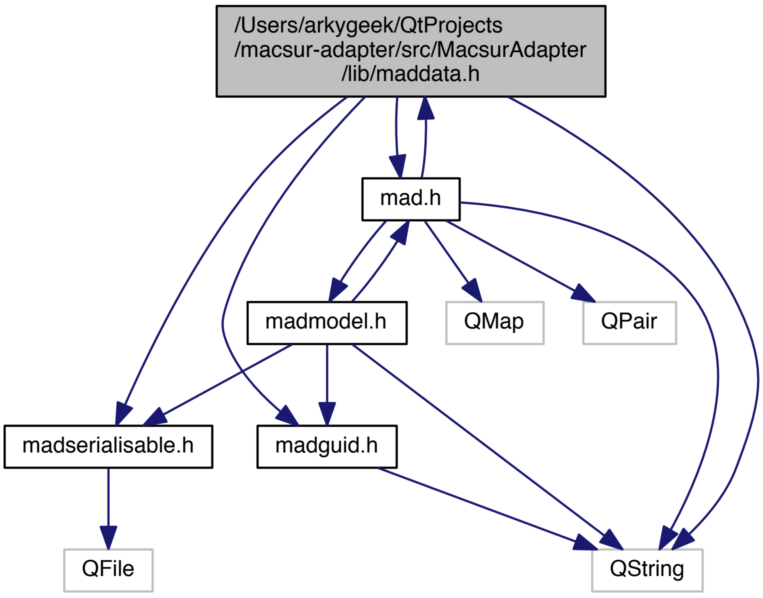
Include dependency graph for maddata.cpp:



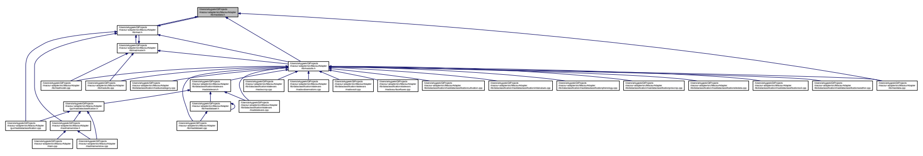
## 7.33 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddata.h File Reference

```
#include "madserialisable.h"
#include "madguid.h"
#include "mad.h"
#include <QString>
```

Include dependency graph for maddata.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class MadData

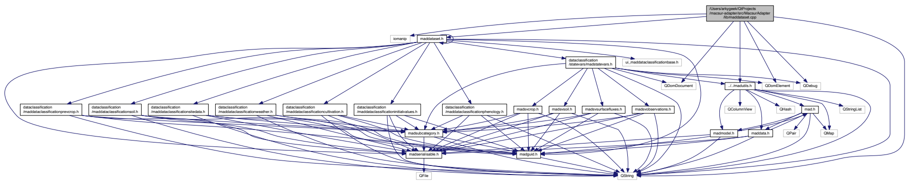
  *The MadData class.*

## 7.34 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddataset.cpp File Reference

```
#include <iomanip>
```

```
#include "maddataset.h"
#include "madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

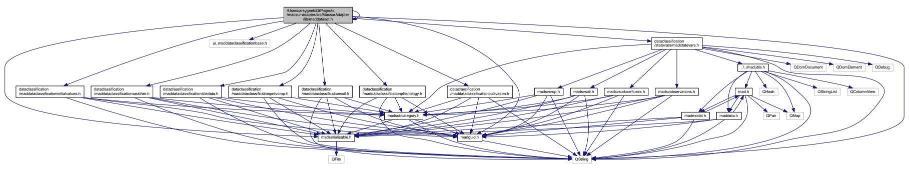
Include dependency graph for maddataset.cpp:



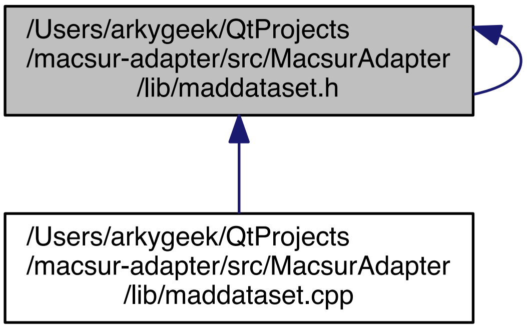
## 7.35 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/maddataset.h File Reference

```
#include "madguid.h"
#include "madserialisable.h"
#include "ui_maddataclassificationbase.h"
#include "maddataset.h"
#include "dataclassification/maddataclassificationcultivation.h"
#include "dataclassification/maddataclassificationinitialvalues.h"
#include "dataclassification/maddataclassificationphenology.h"
#include "dataclassification/maddataclassificationprevcrop.h"
#include "dataclassification/maddataclassificationsitedata.h"
#include "dataclassification/maddataclassificationsoil.h"
#include "dataclassification/maddataclassificationweather.h"
#include "dataclassification/statevars/madstatevars.h"
#include <QString>
```

Include dependency graph for maddataset.h:



This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/maddataset.cpp

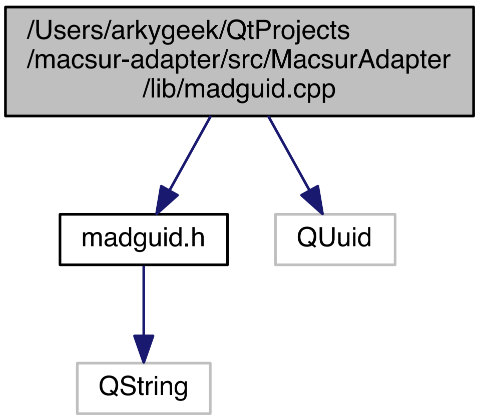
### Classes

- class MadDataset

## 7.36 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madguid.cpp File Reference

```
#include "madguid.h"
#include <QUuid>
```

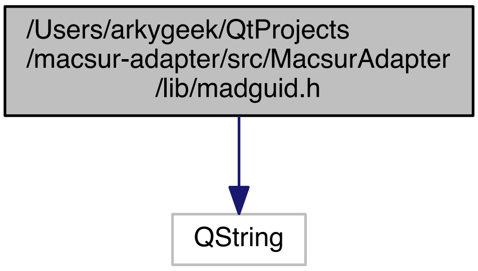
Include dependency graph for madguid.cpp:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/madguid.cpp
madguid.h
QUuid
QString

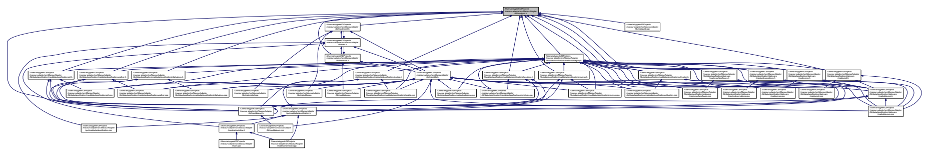
## 7.37 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madguid.h File Reference

```
#include <QString>
```

Include dependency graph for madguid.h:



This graph shows which files directly or indirectly include this file:



### Classes

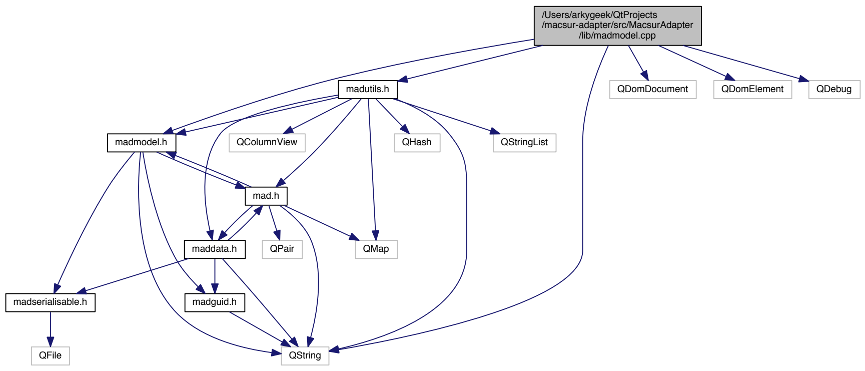
- class MadGuid

  *The MadGuid class An abstract base class that has a Globally Unique Identifier (GUID) to represent a unique instance.*

## 7.38 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madmodel.cpp File Reference

```
#include "madmodel.h"
#include "madutils.h"
#include <QString>
#include <QDomDocument>
#include <QDomElement>
#include <QDebug>
```

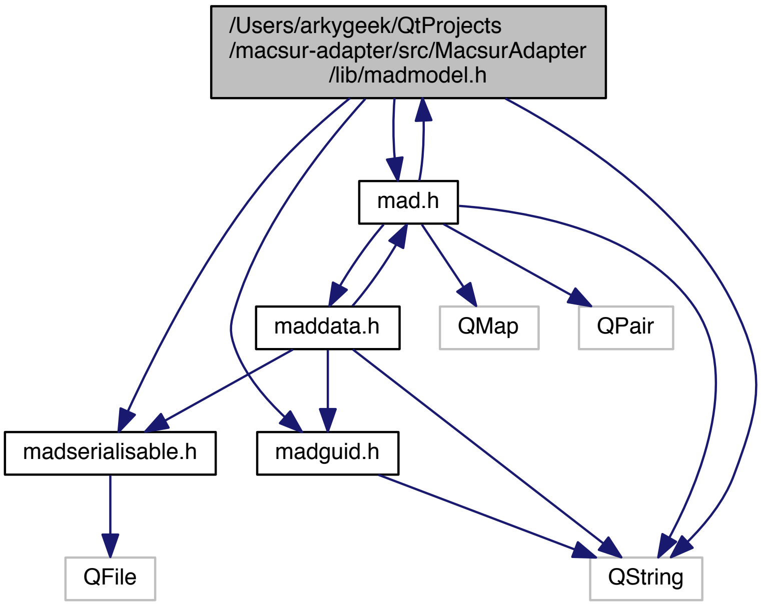
Include dependency graph for madmodel.cpp:



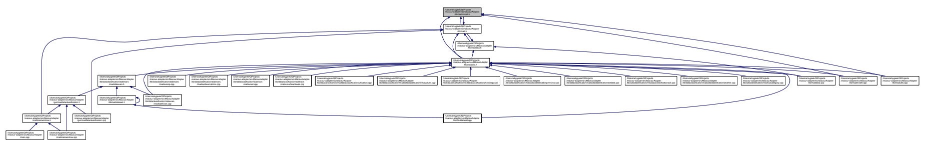
## 7.39 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madmodel.h File Reference

```
#include "madserialisable.h"
#include "madguid.h"
#include "mad.h"
#include <QString>
```

Include dependency graph for madmodel.h:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/madmodel.h
mad.h
maddata.h
QMap
QPair
madserialisable.h
madguid.h
QFile
QString

This graph shows which files directly or indirectly include this file:



### Classes

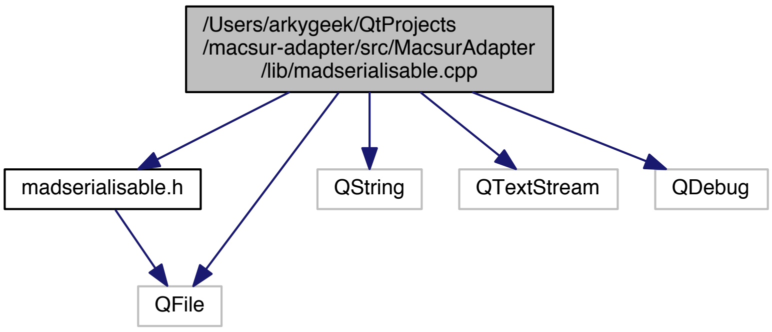
- class MadModel

  *The MadModel class, to represent a ModelTheme.*

## 7.40 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madserialisable.cpp File Reference

```
#include "madserialisable.h"
#include <QFile>
#include <QString>
#include <QTextStream>
#include <QDebug>
```

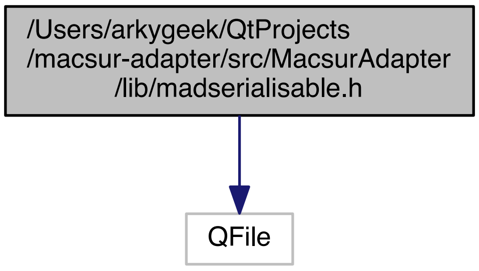
Include dependency graph for madserialisable.cpp:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/madserialisable.cpp
madserialisable.h
QString
QTextStream
QDebug
QFile

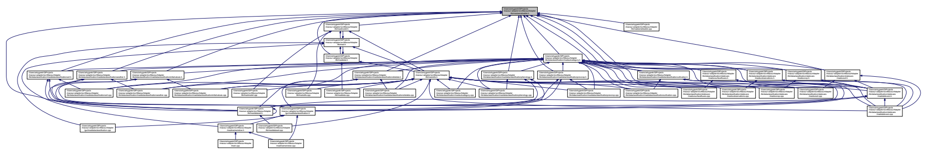
## 7.41 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madserialisable.h File Reference

```
#include <QFile>
```

Include dependency graph for madserialisable.h:



This graph shows which files directly or indirectly include this file:



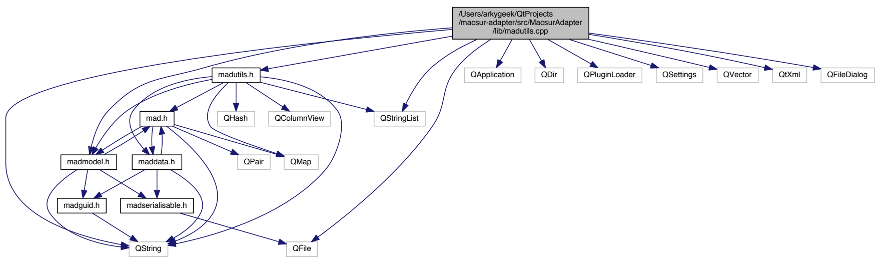
### Classes

- class MadSerialisable

## 7.42 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madutils.cpp File Reference

```
#include "madutils.h"
#include "madmodel.h"
#include <QApplication>
#include <QDir>
#include <QFile>
#include <QPluginLoader>
#include <QSettings>
#include <QString>
#include <QStringList>
#include <QVector>
#include <QtXml>
#include <QFileDialog>
```

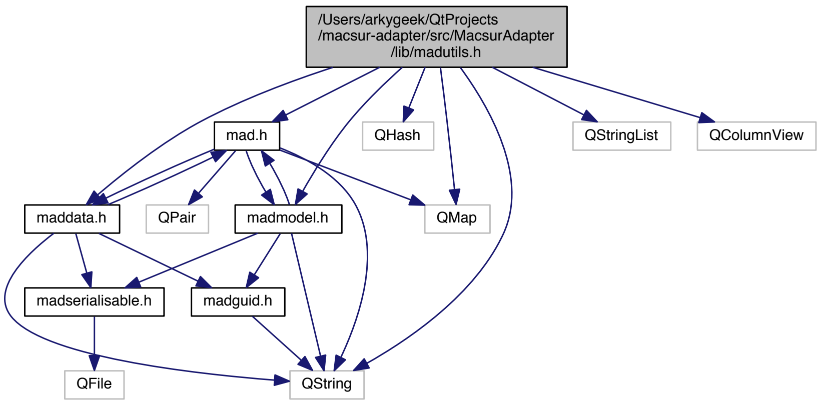
Include dependency graph for madutils.cpp:



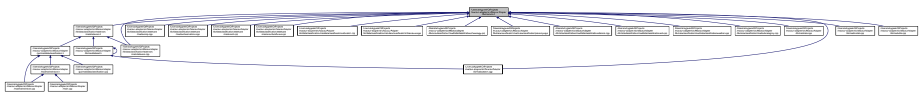
## 7.43 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madutils.h File Reference

```
#include "mad.h"
#include "madmodel.h"
#include "maddata.h"
#include <QHash>
#include <QMap>
#include <QString>
#include <QStringList>
#include <QColumnView>
```

Include dependency graph for madutils.h:



This graph shows which files directly or indirectly include this file:

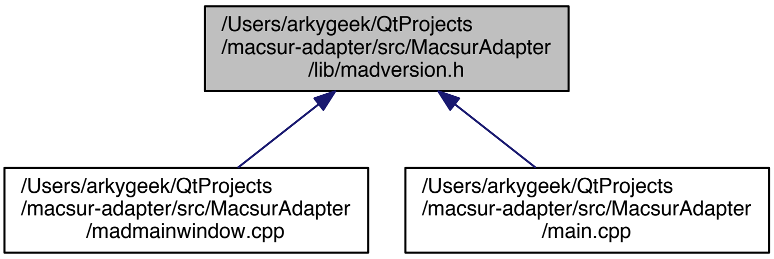


## Classes

- class MadUtils

## 7.44 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/lib/madversion.h File Reference

This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /lib/madversion.h
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /madmainwindow.cpp
/Users/arkygeek/QtProjects /macsur-adapter/src/MacsurAdapter /main.cpp

### Macros

- #define VERSION "0.1"

### 7.44.1 Macro Definition Documentation

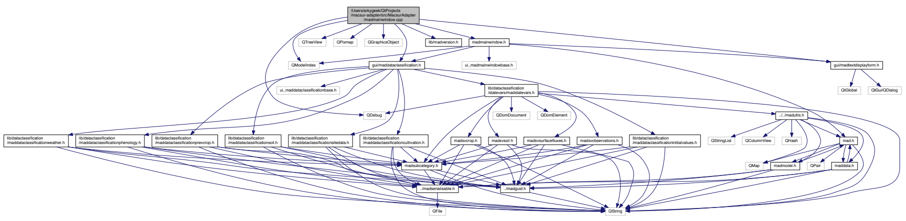
#### 7.44.1.1 #define VERSION "0.1"

Definition at line 23 of file madversion.h.

## 7.45 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/madmainwindow.cpp File Reference

```
#include <QModelIndex>
#include <QDebug>
#include <QTreeView>
#include <QPixmap>
#include <QGraphicsObject>
#include "madmainwindow.h"
#include "lib/madversion.h"
#include "gui/maddataclassification.h"
#include "gui/madtextdisplayform.h"
```

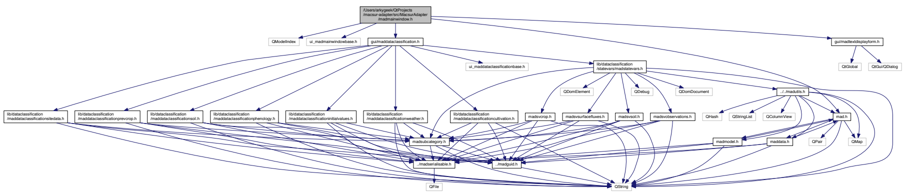
Include dependency graph for madmainwindow.cpp:



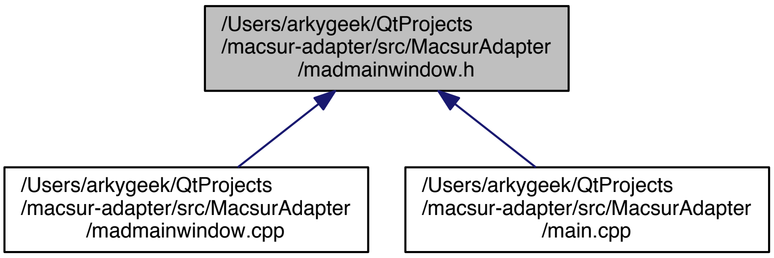
## 7.46 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/madmainwindow.h File Reference

```
#include <QModelIndex>
#include "ui_madmainwindowbase.h"
#include "gui/maddataclassification.h"
#include "gui/madtextdisplayform.h"
#include "lib/mad.h"
```

Include dependency graph for madmainwindow.h:



This graph shows which files directly or indirectly include this file:


/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/madmainwindow.h
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/madmainwindow.cpp
/Users/arkygeek/QtProjects
/macsur-adapter/src/MacsurAdapter
/main.cpp

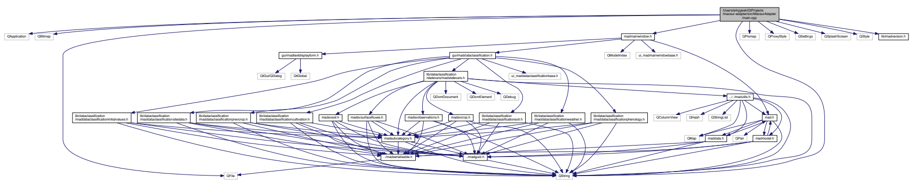
### Classes

- class MadMainWindow

## 7.47 /Users/arkygeek/QtProjects/macsur-adapter/src/MacsurAdapter/main.cpp File Reference

```
#include <QApplication>
#include <QBitmap>
#include <QFile>
#include <QPixmap>
#include <QProxyStyle>
#include <QSettings>
#include <QSplashScreen>
#include <QString>
#include <QStyle>
#include "madmainwindow.h"
#include "lib/madversion.h"
```

Include dependency graph for main.cpp:



## Functions

- int main (int argc, char ∗argv[])
- bool bundleclicked (int argc, char ∗argv[])

## 7.47.1 Function Documentation

### 7.47.1.1 bool bundleclicked ( int *argc*, char ∗ *argv[]* )

Definition at line 77 of file main.cpp.

```
78 {
79   return ( argc > 1 && memcmp(argv[1], "-psn_", 5) == 0 );
80 }
```

### 7.47.1.2 int main ( int *argc*, char ∗ *argv[]* )

Definition at line 47 of file main.cpp.

```
48 {
49     QApplication a(argc, argv);
50
51 #ifdef Q_WS_WIN
52     //for windows lets use plastique syle!
53   QApplication::setStyle(new QPlastiqueStyle);
54 #endif
55
56 #ifdef Q_OS_MACX
57   QString bundledQtCore(QCoreApplication::applicationDirPath().append
58                        ("/lib/QtCore.framework"));
59   if (QFile::exists(bundledQtCore))
60   {
61     QCoreApplication::setLibraryPaths
62         (QStringList(QCoreApplication::applicationDirPath()));
63   }
64 #endif
65
66     MadMainWindow w;
67     w.show();
68
69     return a.exec();
70 }
```

# Index