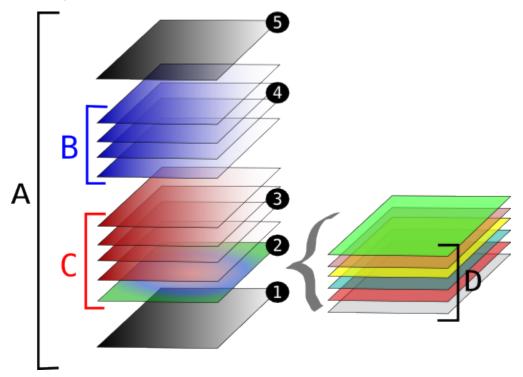
Map2D structure

JDefaultMap2D structure



A layered structure

- A: The all set of layer will result in interactive widget map
- B: The Decoration group, you can control all layer in here
- C: The MapDecoration group, this one only accesible within the class, so you'll have extend the class
- D: The context
- 5: The top Decoration is the InformationDecoration, you can give him some text to show up if needed. He also show up the small "redrawing" message when the map is in rendering state.
- 4: The user Decoration, in this one you will found the scalebar layer, the minimap and other interactive layers
- 3: These are the most important Decoration, you found in them the edition/selection Decoration and others importants ones
- 2: This layer is the result of the RenderingStrategy of the map, this is the layer in which all your maplayers are rendered
- 1 : And the background Decoration, you can replace it to change the background color or draw a grid or even a nasty image 🤩



UML

RenderingStrategy

{From strategy}

Attributes

Operations public Coordinate toMapCoord(int mx, int my) public Point toComponentCoord(Coordinate coord) public BufferedImage createBufferImage(MapLayer layer) public BufferedImage createBufferImage(MapContext context) public BufferedImage getBufferImage() public void setContext(MapContext context) public MapContext getContext() public void setMapArea(Envelope area) public Envelope getMapArea() public void refresh() public JComponent getComponent() public void addStrategyListener(StrategyListener listener) public void removeStrategyListener(StrategyListener listener) public StrategyListener[0..*] getStrategyListeners() public void setAutoRefreshEnabled(boolean refresh) public boolean is AutoRefresh() public boolean isPainting()

<<interface>> StrategyListener {From listener}

Attributes

Operations

public void setRendering(boolean rendering)
public void mapAreaChanged(Map2DMapAreal
public void mapContextChanged(Map2DConte

RENDERING

<<interface>> Map2D

(From map2d)

Attributes

Operations public void setRenderingStrategy(RenderingStrategy strategy) public RenderingStrategy getRenderingStrategy() public Component getComponent() public void addMap2DListener(Map2DListener listener) public void removeMap2DListener(Map2DListener listener) public Map2DListener[0..*] getMap2DListeners() public void setActionState(ACTION_STATE state) public ACTION_STATE getActionState() public void setInformationDecoration(InformationDecoration info) public InformationDecoration getInformationDecoration() public void setBackgroundDecoration(MapDecoration back) public MapDecoration getBackgroundDecoration() public void addDecoration(MapDecoration deco) public void addDecoration(intindex, MapDecoration deco) public int getDecorationIndex(MapDecoration deco) public void removeDecoration(MapDecoration deco) public MapDecoration[0."] getDecorations()

<<interface>> MapDecoration

Attributes

Operations
public void refresh()
public void setMap2D(Map2D map)
public Map2D getMap2D()
public JComponent geComponent()

Map2D

<<interface>> Map 2D Listener

{ From listener }

Attributes Operations

public void mapStrategyChanged(Map2DEvent mapEvent) public void mapActionStateChanged(Map2DEvent mapEvent)

Map 2D Event

{ From event }

Attributes

private RenderingStrategy oldstrategy private RenderingStrategy newstrategy

Operations
public Map2DEvent(Map2D map, ACTION_STATE oldaction, ACTION_STATE newaction, RenderingStrategy strategy) public Map2DEvent(Map2D map, ACTION_STATE action, RenderingStrategy oldstrategy, RenderingStrategy newstrategy) public ACTION_STATE getPreviousState() public ACTION_STATE getNewState() public RenderingStrategy getNewStrategy() public RenderingStrategy getPreviousStrategy()

Exemples

• An ImageOverLayer as BackLayer :

