**setDensityMaskRegion in FS25**

**Function Parameters and Description**

The setDensityMaskRegion function is a global engine function in Farming Simulator (FS) that defines a **masked region** on a density map. It is typically used to isolate or highlight a specific layer (channel) of terrain data over a given area. The function signature and parameters are as follows:

* **densityMapId** (integer): The ID of the density map to apply the mask on. In FS, this is often a *terrain detail* or foliage layer ID (e.g. g\_currentMission.terrainDetailId) which contains multiple terrain “info layers” (such as crop types, growth stages, soil states) (FS25\_BetterMinimap.lua).
* **startX** (integer): The X-coordinate (column) where the mask region begins on the density map grid. Typically measured in density map pixels (0 is one edge of the map).
* **startZ** (integer): The Z-coordinate (row) where the mask region begins on the density map grid. (In FS, X,Z correspond to horizontal coordinates on the map’s 2D grid.)
* **width** (integer): The width of the region (in density map units) to mask, starting from startX. For example, on a 2048×2048 map, a width of 2048 covering startX=0 spans the entire map’s width (FS25\_BetterMinimap.lua).
* **height** (integer): The height of the region (in density map units) to mask, starting from startZ. Similarly, height=2048 with startZ=0 would cover the full map’s height.
* **maskTypeIndex** (integer): The index of the terrain **density layer** (or “channel”) to use as the mask. This corresponds to a particular sub-layer within the density map. For example, an index can represent the fruit type layer, growth state layer, fertilizer state, etc. The terrain’s multi-layer setup assigns each category a type index (often defined in the map’s I3D as densityMapTypeIndex). Using a specific index here isolates that layer. For instance, 1 might target the fruit type layer, 2 the growth stage layer, and so on (FS25\_BetterMinimap.lua).

**Behavior:** Calling setDensityMaskRegion essentially tells the engine to **mask out** (focus on) the specified region and layer on the given density map. It doesn’t by itself modify the map; rather, it sets an internal mask that subsequent operations or renderings will use. In practice, this is used to filter or highlight certain data. The function likely returns an integer count of cells affected (or 0) when setting the mask, though its primary effect is on engine state (often no direct return value is used in scripts, as the call’s purpose is to set up the mask).

**Interaction with Terrain Density Layers**

Terrain in FS is represented by density maps where multiple “layers” of information are packed into one or more textures (for efficiency). For example, a single **terrain detail map** might encode: crop *type*, crop *growth stage*, plowing state, lime state, fertilizer level, and weed presence – each stored in certain bits or channels of the density map. The maskTypeIndex parameter of setDensityMaskRegion corresponds to one of these layers (also known as a *densityMapTypeIndex* in the map’s configuration).

Using setDensityMaskRegion with a specific index effectively **filters the density map to that layer** within the defined region. Only data from that layer will be considered by subsequent engine calls or rendering of overlays. In other words, it creates a mask so that operations (like drawing the minimap overlay or running a density modification) apply **only where the chosen layer has data** (FS25\_BetterMinimap.lua) (FS25\_BetterMinimap.lua). This is particularly useful for displaying or processing one aspect of terrain data at a time. For example:

* **Fruit Type Layer Mask:** Using maskTypeIndex = 1 (example index for fruit/crop type) will mask the region to show different crop types planted on fields, each crop being identified by its type value in that layer.
* **Growth Stage Layer Mask:** Using maskTypeIndex = 2 might isolate the growth stage layer, allowing one to highlight how far along crops are in growth within the region (e.g. which fields are ready to harvest vs recently seeded) (FS25\_BetterMinimap.lua).
* **Soil/Field State Layers:** Higher indices often correspond to soil composition states – e.g. cultivated vs plowed, fertilized stages, need-plowing flag, need-lime flag, etc. Masking those layers lets the engine highlight fields that are fertilized or require plowing, etc. (FS25\_BetterMinimap.lua).

Under the hood, the engine knows which bits of the density map correspond to the given type index and will use that as a filter. This function does **not** permanently alter the density data; it only sets the mask for subsequent operations. Typically, after you finish the masked operation (such as drawing an overlay or performing a calculation), you might clear or change the mask (for example, by setting a different mask or resetting to an unmasked state). If you need to remove the mask, calling the function on a region/layer that covers nothing (or switching back to a default state) would effectively stop masking.

**Important:** setDensityMaskRegion is often used in conjunction with other density map functions. For instance, the engine also provides functions like setDensityCompareParams and setDensityParallelogram/setDensityMaskedParallelogram for editing density maps using masks and conditions. In the context of terrain overlays (visualizing the data), the mask function is usually paired with internal rendering routines rather than manual Lua drawing. In FS25’s engine (GIANTS Engine 10), these masks feed into the system that generates map overlays.

**Usage Examples in FS25 Scripts**

**Minimap Overlays:** One common use of setDensityMaskRegion in FS25 is to toggle what the in-game **minimap** displays. By masking different layers of the terrain detail, scripters can mimic the various filters available in the main map (fruit types, growth, soil status, etc.) on the HUD minimap. For example, the *Better Minimap* mod for FS25 uses this function to switch the minimap’s view mode:

* To show **Fruit Types** on the minimap, the mod calls:
* local terrainDetail = g\_currentMission.terrainDetailId
* setDensityMaskRegion(terrainDetail, 0, 0, 2048, 2048, 1)

This masks the entire map (0,0 to 2048,2048) to the layer with index 1 (which corresponds to crop types) (FS25\_BetterMinimap.lua). The result is that the minimap will color each field by the crop planted.

* To show **Growth Stage** instead, it uses index 2 for the same region:
* setDensityMaskRegion(terrainDetail, 0, 0, 2048, 2048, 2)

Now the minimap highlights growth stages (for instance, ready-to-harvest fields might be shown in a particular color) (FS25\_BetterMinimap.lua).

* Likewise, other calls in the mod use indices 3 through 6 to display **Soil State**, **Fertilizer**, **Plowing**, or **Lime requirements** respectively (FS25\_BetterMinimap.lua). Each of these layers is a part of the combined terrainDetailId map. By setting the mask region to the full map and choosing the appropriate layer index, the minimap overlay is refreshed to show that specific information.

In these examples, terrainDetail is the density map for fields, and the numbers 0,0 and 2048,2048 cover the full size of a standard map. The mod cycles through modes by calling setDensityMaskRegion with different maskTypeIndex values on the same region (FS25\_BetterMinimap.lua) (FS25\_BetterMinimap.lua). Internally, the game’s HUD system then renders the minimap using the active mask, so the player sees the colored overlay corresponding to the chosen filter (e.g., fruit types or growth stages).

**Scripted Field Operations:** Aside from visual overlays, a script could use setDensityMaskRegion to limit a subsequent terrain edit to a particular layer. For example, if a mod wanted to remove only a specific crop from an area or count how many pixels of a certain state exist, it could set a mask and then use a function like setDensityParallelogram or getDensityRegion to act only on that masked layer. However, in practice, there are other functions like setDensityMaskParams and setDensityCompareParams that provide more fine-grained control for such editing. The setDensityMaskRegion is most directly useful when generating map overlays or doing broad masks by layer.

FS25’s engine actually includes built-in mechanisms to generate these overlays (via the **MapOverlayGenerator** class). In fact, the game uses density map data to create the colored maps you see in the menu. The existence of setDensityMaskRegion allows modders to tap into a similar effect for the HUD. The GIANTS Engine documentation notes that the engine can provide “density map based data overlays on top of an in-game map” ([GDN {GIANTS Developer Network}](https://gdn.giants-software.com/documentation_scripting_fs19.php?version=script&category=15&class=177#:~:text=,game%20map)) – which is exactly what masking a terrain layer achieves.

**Relevant Documentation and References**

* **GIANTS Developer Network (GDN):** The official scripting reference for Farming Simulator (FS) provides details on engine functions. As of FS25, the FS22 Lua API documentation is the latest available on GDN ([FS25 Scripting documentation : r/farmingsimulator](https://www.reddit.com/r/farmingsimulator/comments/1hv1la6/fs25_scripting_documentation/#:~:text=If%20Giants%20has%20released%20any,rather%20than%20a%20comprehensive%20sdk)). It includes functions for terrain detail manipulation. While setDensityMaskRegion may not be explicitly listed in the public docs, related functions and concepts (like density maps and map overlays) are documented. The **MapOverlayGenerator** class in FS22/FS25, for example, is described as providing *“density map based data overlays”* for the in-game map ([GDN {GIANTS Developer Network}](https://gdn.giants-software.com/documentation_scripting_fs19.php?version=script&category=15&class=177#:~:text=,game%20map)), which underlies this function’s usage. (Developers can refer to GDN for FS22 and expect FS25 to have similar functions.)
* **FS25\_BetterMinimap Mod (2025)** – *by SupremeClicker*: This mod demonstrates practical use of setDensityMaskRegion. The code sets the mask on g\_currentMission.terrainDetailId for different layers (fruit, growth, etc.) based on user-selected mode (FS25\_BetterMinimap.lua) (FS25\_BetterMinimap.lua). It’s an excellent reference for how to apply the function to achieve minimap filtering. Reviewing such script mods is often recommended by GIANTS and the community to learn correct usage.
* **FS17 BetterMinimap (Original by jDanek)**: Earlier versions of the Better Minimap (for FS17) achieved similar results using other techniques. While FS17’s engine had similar density map functions, modders sometimes resorted to toggling the game’s map overview states via UI code. FS25’s approach with setDensityMaskRegion is more direct. The evolution of this mod across game versions shows how setDensityMaskRegion can simplify the implementation for newer FS versions.

In summary, setDensityMaskRegion is used in FS25 to focus on a specific terrain layer within a region, most notably for drawing colored overlays on the map or minimap. Understanding the terrain’s layer indices (mask type indices) is key – these indices map to what kind of data you are isolating (crops, growth, soil info). By consulting Giants’ documentation and looking at mods that use this function, one can get the exact parameters and see it in action. The references above (GDN and the BetterMinimap script) provide guidance on how the function works and how to use it effectively (FS25\_BetterMinimap.lua) (FS25\_BetterMinimap.lua).