### **Graph API (gapi module)**

The G-API (Graph API) is a special OpenCV module introduced in OpenCV 4.0. Unlike usual functions, it's more like a framework for building image processing pipelines that can run really efficiently on different hardware.

#### Why G-API is Useful

* Majority of OpenCV functions are optimized individually (like blur or resize), but G-API lets us optimize whole pipelines of operations automatically.
* It captures all steps (like resize -> gray -> blur -> canny) in a graph, which lets OpenCV do smart optimizations like tiling (processing smaller chunks), improving memory usage and speed.
* We only describe what should happen, not how. G-API handles the rest — this makes code cleaner and easier to move to new platforms like GPUs or special accelerators.

#### How G-API Works

* First, a graph of operations is declared in code (using cv::gapi::resize, cv::gapi::blur, cv::gapi::Canny, etc.) treating inputs as cv::GMat, not real data.
* Then a cv::GComputation object captures this graph. No actual processing happens yet — it's just the structure.
* Later, when data is ready (like a live video frame), G-API compiles the graph based on available backends and runs it efficiently.

#### Backends Supported

* CPU backend — runs using classic OpenCV code; good for quick prototyping.
* Fluid backend — better memory usage and cache behavior, especially helpful for big images.
* OpenCL (GPU) backend — runs on GPUs via OpenCL-capable devices.
* OpenVINO backend — uses Intel’s optimized inference engine for deep learning part of pipelines.
* ONNX backend — runs ONNX models via ONNX Runtime inside G-API.

#### Real-World Example

One tutorial shows how to build a video pipeline that: decodes video, runs face detection, age/gender/emotion inference, and shows results, all inside one G-API graph. G-API automatically handles streaming and optimization on supported hardware.