Here’s a comprehensive list of GStreamer **source elements** categorized by their functionality:

**1. Video Sources**

These elements generate or capture video streams.

| **Source Name** | **Description** |
| --- | --- |
| videotestsrc | Generates test video patterns (useful for debugging). |
| v4l2src | Captures video from a V4L2 device (e.g., webcam, capture card). |
| ximagesrc | Captures video from the X11 screen. |
| gltestsrc | Generates test video using OpenGL. |
| decklinkvideosrc | Captures video from a Blackmagic DeckLink device. |
| kmssink | Captures video from DRM/KMS devices (Linux display systems). |

**2. Audio Sources**

These elements generate or capture audio streams.

| **Source Name** | **Description** |
| --- | --- |
| audiotestsrc | Generates test audio patterns (e.g., sine waves). |
| alsasrc | Captures audio from ALSA devices (Linux). |
| pulsesrc | Captures audio from PulseAudio. |
| oss4src | Captures audio from OSS (Open Sound System). |
| wasapisrc | Captures audio from WASAPI (Windows Audio Session API). |
| jackaudiosrc | Captures audio from the JACK audio system (professional audio workflows). |

**3. File Sources**

These elements read data from files.

| **Source Name** | **Description** |
| --- | --- |
| filesrc | Reads raw data from a file. |
| multifilesrc | Reads data from multiple files (sequentially or based on a pattern). |
| splitfilesrc | Reads fragmented or split files. |

**4. Network Sources**

These elements capture data from network streams.

| **Source Name** | **Description** |
| --- | --- |
| udpsrc | Receives data from a UDP network socket. |
| tcpclientsrc | Receives data from a TCP server. |
| souphttpsrc | Downloads data over HTTP using the Soup library. |
| rtpsrc | Receives RTP (Real-Time Protocol) streams. |
| rtspsrc | Receives data from an RTSP server (used for cameras and live streams). |
| webrtcbin | Receives WebRTC streams for real-time communication. |
| srtclientsrc | Receives data using the SRT (Secure Reliable Transport) protocol. |

**5. Application/Custom Sources**

These elements provide data from applications or custom sources.

| **Source Name** | **Description** |
| --- | --- |
| appsrc | Allows applications to push custom data into a pipeline. |
| intervideosrc | Shares video data between pipelines using shared memory. |
| interaudiosrc | Shares audio data between pipelines using shared memory. |
| shmsrc | Reads data from shared memory. |

**6. Streaming and Media Sources**

These elements handle media streaming and specialized protocols.

| **Source Name** | **Description** |
| --- | --- |
| playbin | High-level source that handles both audio and video playback automatically. |
| urisourcebin | Reads data from URIs (e.g., HTTP, local files, etc.). |
| dvbbasebin | Captures data from a DVB (Digital Video Broadcasting) source. |
| hlsdemux | Captures streams from HLS (HTTP Live Streaming) playlists. |

**7. Debugging and Test Sources**

These elements are useful for testing and debugging.

| **Source Name** | **Description** |
| --- | --- |
| videotestsrc | Generates test video patterns. |
| audiotestsrc | Generates test audio signals. |
| fakesrc | Generates empty data streams for testing. |

**8. Hardware and Specialized Sources**

These elements capture or generate data from specific hardware or specialized use cases.

| **Source Name** | **Description** |
| --- | --- |
| decklinkaudiosrc | Captures audio from a Blackmagic DeckLink device. |
| decklinkvideosrc | Captures video from a Blackmagic DeckLink device. |
| qtiqmmfsrc | Captures video using Qualcomm’s camera framework (Android). |
| nvarguscamerasrc | Captures video from NVIDIA Jetson platforms using the Argus framework. |

**Summary of Categories**

1. **Video Sources**: Generate or capture video streams.
2. **Audio Sources**: Generate or capture audio streams.
3. **File Sources**: Read data from files.
4. **Network Sources**: Capture data from network streams.
5. **Application/Custom Sources**: Provide data from applications or shared memory.
6. **Streaming/Media Sources**: Handle media streaming or playback.
7. **Debugging/Test Sources**: Useful for testing pipelines.
8. **Hardware/Specialized Sources**: Capture data from specific hardware or frameworks.

GStreamer sinks are categorized based on the type of output or destination they handle. Here's a comprehensive list of sink types with examples for each category:

**1. Screen/Display Sinks**

These sinks are used to render video onto a screen or display.

| **Sink Name** | **Description** |
| --- | --- |
| autovideosink | Automatically selects the best available video sink for the system. |
| xvimagesink | Renders video using the X-Video extension (X11 systems). |
| glimagesink | Renders video using OpenGL (hardware-accelerated). |
| waylandsink | Renders video on Wayland-based systems. |
| cacasink | Displays video in ASCII art (for fun/debugging). |
| qt5glsink | Renders video using a Qt5 OpenGL widget. |

**2. Audio Output Sinks**

These sinks handle audio playback.

| **Sink Name** | **Description** |
| --- | --- |
| autoaudiosink | Automatically selects the best available audio sink. |
| alsasink | Outputs audio to ALSA devices (Linux). |
| pulsesink | Outputs audio to PulseAudio. |
| osssink | Outputs audio using OSS (Open Sound System). |
| directsoundsink | Outputs audio using DirectSound (Windows). |
| wasapisink | Outputs audio using WASAPI (Windows Audio Session API). |
| jackaudiosink | Outputs audio to JACK (for professional audio workflows). |

**3. File Output Sinks**

These sinks write data to a file.

| **Sink Name** | **Description** |
| --- | --- |
| filesink | Writes raw or encoded data to a file. |
| multifilesink | Writes data to multiple files, creating a new file for each buffer. |
| splitmuxsink | Writes segmented files (useful for adaptive streaming). |

**4. Network Sinks**

These sinks transmit data over a network.

| **Sink Name** | **Description** |
| --- | --- |
| udpsink | Sends data over UDP to a network address. |
| tcpserversink | Sends data over a TCP connection to a client. |
| rtpsink | Sends RTP (Real-Time Protocol) streams over the network. |
| hlssink | Creates HLS (HTTP Live Streaming) segments and playlists. |
| srtserversink | Sends data over SRT (Secure Reliable Transport) protocol. |

**5. Storage and Device Sinks**

These sinks output data to hardware devices or storage locations.

| **Sink Name** | **Description** |
| --- | --- |
| v4l2sink | Outputs video to a V4L2 device (e.g., webcam or external device). |
| fbdevsink | Displays video directly on the framebuffer device. |
| decklinksink | Outputs video to a Blackmagic DeckLink device. |

**6. Application/Programmatic Sinks**

These sinks allow the application to process data programmatically.

| **Sink Name** | **Description** |
| --- | --- |
| appsink | Sends data to the application for custom processing. |
| intervideosink | Shares video data between pipelines using shared memory. |
| interaudiosink | Shares audio data between pipelines using shared memory. |

**7. Debugging and Testing Sinks**

These sinks are useful for testing and debugging pipelines.

| **Sink Name** | **Description** |
| --- | --- |
| fakesink | Discards data (useful for testing pipelines without rendering). |
| identity | Passes data through without modification (useful for debugging). |
| cacasink | Displays video as ASCII art (also in Screen/Display category). |

**8. Media and Streaming Sinks**

These sinks are specialized for media streaming or handling.

| **Sink Name** | **Description** |
| --- | --- |
| shmsink | Shares data between processes using shared memory. |
| souphttpsink | Sends data over HTTP using the Soup library. |
| rtmpsink | Sends data to an RTMP server (commonly used for live streaming). |

**Summary of Categories**

1. **Screen/Display Sinks**: Render video to displays.
2. **Audio Output Sinks**: Play audio.
3. **File Output Sinks**: Write data to files.
4. **Network Sinks**: Transmit data over the network.
5. **Storage/Device Sinks**: Output to hardware or storage devices.
6. **Application/Programmatic Sinks**: Allow custom processing in applications.
7. **Debugging/Testing Sinks**: Useful for testing pipelines.
8. **Media/Streaming Sinks**: Handle streaming and specialized protocols.

**Example Use Cases:**

* **Screen Rendering**: Use autovideosink to render video automatically.
* **Save Video to File**: Use filesink to save raw or encoded video streams.
* **Network Streaming**: Use udpsink or rtmpsink for streaming over the network.
* **Custom Processing**: Use appsink to process video frames programmatically in your application.