

# PAL API Reference

Generated from public `Protocol` declarations and source docstrings. Each description is maintained beside its source signature. Regenerate with `python Documentation/generators/generate_api_reference.py` ; use `--check` to detect drift. Consult the PAL Guide for worked usage and Architecture and Design for implementation rationale.

Axis shorthand: `...` after an X/x-specific name includes its Y/y and Z/z variants. Each variant remains a separate callable or property.

## Factories and utilities

| Function                                                                  | Description                                                                                                                                                                              |
|---------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>FastMode()</code>                                                   | Select PAL fast mode before constructing any PAL object or window; the process-wide mode cannot be changed after construction.                                                           |
| <code>Seed(seed)</code>                                                   | Seed PAL's shared random-number stream; repeatability also depends on the sequence of random calls.                                                                                      |
| <code>Random()</code>                                                     | Draw a uniform random value from PAL's shared random-number stream.                                                                                                                      |
| <code>RandInt(max)</code>                                                 | Draw an integer from PAL's random stream in the supported range below the positive bound <code>max</code> .                                                                              |
| <code>AwaitWindows(timeout=30)</code>                                     | Wait for PAL windows that are still initializing; raises if a window exits early or initialization times out.                                                                            |
| <code>ColorScale(colors, value)</code>                                    | Map a scalar value to an RGB color by interpolating through the supplied color scale.                                                                                                    |
| <code>NewIList()</code>                                                   | Create an empty PAL integer list usable from Python and supported compiled functions.                                                                                                    |
| <code>MooreHood(dim, excludeCenter=False)</code>                          | Build relative integer offsets for a Moore neighborhood in <code>dim</code> dimensions; optionally exclude the center offset.                                                            |
| <code>VonNeumannHood(dim, excludeCenter=False)</code>                     | Build relative integer offsets for a Von Neumann neighborhood in <code>dim</code> dimensions; optionally exclude the center offset.                                                      |
| <code>CircleHood(dim, rad, excludeCenter=False)</code>                    | Build relative integer offsets within radius <code>rad</code> in <code>dim</code> dimensions; optionally exclude the center offset.                                                      |
| <code>NewMultinomial(other=None)</code>                                   | Create a PAL multinomial/binomial sampler, optionally reusing configuration from <code>other</code> .                                                                                    |
| <code>NewGrid(dimensions, dtype)</code>                                   | Create a typed lattice grid with the requested dimensions and element dtype.                                                                                                             |
| <code>NewAgentGrid(dimensions, numAgentProps=0, isStackable=False)</code> | Create a grid for individual agents; <code>numAgentProps</code> sets the number of per-agent properties and <code>isStackable</code> controls whether multiple agents may occupy a site. |

| Function                                           | Description                                                                                                                                                                                     |
|----------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>NewPopGrid(dimensions, capacity=None)</code> | Create a lattice of integer population counts; <code>capacity</code> selects the smallest integer storage type that accommodates the requested per-site count, without enforcing it as a limit. |
| <code>NewPDEgrid(dimensions)</code>                | Create a lattice-backed scalar field for numerical transport operations.                                                                                                                        |

## Multinomial

| Kind   | Declaration                                             | Description                                                                                                                             |
|--------|---------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| method | <code>Binomial(self, n: int, p: float) -&gt; int</code> | Draw a binomial sample with <code>n</code> trials and success probability <code>p</code> using this sampler's configured random stream. |
| method | <code>Setup(self, n: int)</code>                        | Configure the sampler for <code>n</code> trials before repeated sampling.                                                               |
| method | <code>Sample(self, p: float) -&gt; int</code>           | Draw a binomial sample using the sampler's current configuration and success probability <code>p</code> .                               |

## IList

| Kind   | Declaration                                          | Description                                                                                              |
|--------|------------------------------------------------------|----------------------------------------------------------------------------------------------------------|
| method | <code>__len__(self) -&gt; int</code>                 | Return the number of integers currently stored in the list.                                              |
| method | <code>__getitem__(self, index: int) -&gt; int</code> | Return the integer stored at the given list position.                                                    |
| method | <code>Append(self, i: int)</code>                    | Append integer <code>i</code> to the end of the list.                                                    |
| method | <code>Clear(self)</code>                             | Remove all values from the list.                                                                         |
| method | <code>Random(self) -&gt; int</code>                  | Return a randomly selected value from the list; the list must be nonempty.                               |
| method | <code>Shuffle(self)</code>                           | Shuffle the list in place using PAL's random stream.                                                     |
| method | <code>All(self) -&gt; np.ndarray</code>              | Return a detached NumPy array containing the list values.                                                |
| method | <code>Iter(self) -&gt; np.ndarray</code>             | Iterate over the current list values without creating the detached copy returned by <code>All()</code> . |

## AgentGrid

| Kind   | Declaration                          | Description                                     |
|--------|--------------------------------------|-------------------------------------------------|
| method | <code>__len__(self) -&gt; int</code> | Return the number of living agents in the grid. |

| Kind     | Declaration                                                                                      | Description                                                                                                                                                                          |
|----------|--------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| property | <code>xDim...(self) -&gt; int</code>                                                             | Number of lattice sites along x; corresponding Y/Z variants use the same contract.                                                                                                   |
| property | <code>nDims(self) -&gt; int</code>                                                               | Number of spatial dimensions.                                                                                                                                                        |
| property | <code>wrapX...(self) -&gt; bool</code>                                                           | Whether x uses periodic boundary wrapping; corresponding Y/Z variants use the same contract.                                                                                         |
| method   | <code>Box(self, x1: int, x2: int, y1: int=None, y2: int=None, z1: int=None, z2: int=None)</code> | Iterate lattice coordinates in the half-open rectangular region $[x1, x2) \times [y1, y2) \times [z1, z2)$ ; omit unused axes in lower dimensions.                                   |
| method   | <code>Hood(self, hood, x: int, y: int=None, z: int=None, *, unroll: bool=False)</code>           | Apply relative neighborhood offsets at a lattice coordinate, respecting wrapped and nonwrapped boundaries. <code>unroll=True</code> enables compiled-loop expansion where supported. |
| method   | <code>GetPop(self) -&gt; int</code>                                                              | Return the number of living agents in the grid.                                                                                                                                      |
| method   | <code>Alive(self, agent: int) -&gt; bool</code>                                                  | Return whether agent currently identifies a living agent.                                                                                                                            |
| method   | <code>NewAgentSQ(self, x: int, y: int=-1, z: int=-1) -&gt; int</code>                            | Create an agent at the center of a lattice site specified by integer coordinates or a supported linear site index.                                                                   |
| method   | <code>NewAgent(self, x: float, y: float=-1.0, z: float=-1.0) -&gt; int</code>                    | Create an agent at the supplied continuous spatial coordinates.                                                                                                                      |
| method   | <code>Dispose(self, agent: int)</code>                                                           | Remove a living agent from the grid; its handle must not be used as a live agent afterward.                                                                                          |
| method   | <code>I(self, agent: int) -&gt; int</code>                                                       | Return the linear lattice-site index containing the agent.                                                                                                                           |
| method   | <code>XSQ...(self, agent: int) -&gt; int</code>                                                  | Return the agent's integer x site coordinate; corresponding Y/Z variants use the same contract.                                                                                      |
| method   | <code>X...(self, agent: int) -&gt; float</code>                                                  | Return the agent's continuous x coordinate; corresponding Y/Z variants use the same contract.                                                                                        |
| method   | <code>__getitem__(self, key) -&gt; float</code>                                                  | Read a supported per-agent property value.                                                                                                                                           |
| method   | <code>__setitem__(self, key, value: float)</code>                                                | Set a supported per-agent property value for the selected agent/property key.                                                                                                        |
| method   | <code>ToI(self, x: int, y: int=-1, z: int=-1) -&gt; int</code>                                   | Convert lattice coordinates to the corresponding linear site index.                                                                                                                  |
| method   | <code>ItoX...(self, i: int) -&gt; int</code>                                                     | Convert a linear site index to its x coordinate; corresponding Y/Z variants use the same contract.                                                                                   |
| method   | <code>InWrapSQX...(self, value: int) -&gt; int</code>                                            | Wrap an integer x site coordinate when x is periodic; corresponding Y/Z variants use the same contract.                                                                              |
| method   | <code>InWrapX...(self, value: float) -&gt; float</code>                                          | Wrap a continuous x coordinate when x is periodic; corresponding Y/Z variants use the same contract.                                                                                 |

| Kind     | Declaration                                                                                                 | Description                                                                                                                                |
|----------|-------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| method   | <code>DispWrapX(self, x1: float, x2: float) -&gt; float</code>                                              | Return the shortest signed x displacement under periodic wrapping.                                                                         |
| method   | <code>DispWrapY(self, y1: float, y2: float) -&gt; float</code>                                              | Return the shortest signed y displacement under periodic wrapping.                                                                         |
| method   | <code>DispWrapZ(self, z1: float, z2: float) -&gt; float</code>                                              | Return the shortest signed z displacement under periodic wrapping.                                                                         |
| property | <code>counts(self) -&gt; object</code>                                                                      | Read-only occupancy counts for lattice sites; change occupancy through agent creation, movement, and disposal methods.                     |
| method   | <code>LastAgent(self, x: int, y: int=-1, z: int=-1) -&gt; int</code>                                        | Return the last agent handle at the specified lattice site, or the empty-site sentinel when none is present.                               |
| method   | <code>MoveSQ(self, agent: int, x: int, y: int=-1, z: int=-1)</code>                                         | Move an agent to a lattice site specified by integer coordinates or a supported linear site index.                                         |
| method   | <code>Move(self, agent: int, x: float, y: float=-1.0, z: float=-1.0)</code>                                 | Move an agent to continuous coordinates; use <code>MoveSQ</code> when movement should target a lattice site.                               |
| method   | <code>AgentsAt(self, x: int, y: int=-1, z: int=-1)</code>                                                   | Return the agent handle or handles occupying the specified lattice site.                                                                   |
| method   | <code>AgentsInRadius(self, rad: float, x: float, y: float=-1.0, z: float=-1.0, *, exclude: int=None)</code> | Return agents within radius <code>rad</code> of a continuous position; <code>exclude</code> can omit one agent from the results.           |
| method   | <code>All(self, shuffle: bool=False) -&gt; np.ndarray</code>                                                | Return a snapshot of living agent handles, optionally shuffled. The snapshot fixes membership, not later world state or occupancy queries. |

## Grid

| Kind     | Declaration                                | Description                                                                        |
|----------|--------------------------------------------|------------------------------------------------------------------------------------|
| method   | <code>__len__(self) -&gt; int</code>       | Return the number of lattice sites.                                                |
| method   | <code>__getitem__(self, key)</code>        | Read the value or state selected by the supported index or coordinate key.         |
| method   | <code>__setitem__(self, key, value)</code> | Assign the value or state selected by the supported index or coordinate key.       |
| property | <code>xDim...(self) -&gt; int</code>       | Number of lattice sites along x; corresponding Y/Z variants use the same contract. |
| property | <code>nDims(self) -&gt; int</code>         | Number of spatial dimensions.                                                      |

| Kind     | Declaration                                                                                      | Description                                                                                                                                                                             |
|----------|--------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| property | <code>wrapX...(self) -&gt; bool</code>                                                           | Whether x uses periodic boundary wrapping; corresponding Y/Z variants use the same contract.                                                                                            |
| method   | <code>Box(self, x1: int, x2: int, y1: int=None, y2: int=None, z1: int=None, z2: int=None)</code> | Iterate lattice coordinates in the half-open rectangular region $[x1, x2) \times [y1, y2) \times [z1, z2)$ ; omit unused axes in lower dimensions.                                      |
| method   | <code>Hood(self, hood, x: int, y: int=None, z: int=None, *, unroll: bool=False)</code>           | Apply relative neighborhood offsets at a lattice coordinate, respecting wrapped and nonwrapped boundaries.<br><code>unroll=True</code> enables compiled-loop expansion where supported. |
| method   | <code>ToI(self, x: int, y: int=-1, z: int=-1) -&gt; int</code>                                   | Convert lattice coordinates to a linear site index.                                                                                                                                     |
| method   | <code>ItoX...(self, i: int) -&gt; int</code>                                                     | Convert a linear site index to its x coordinate; corresponding Y/Z variants use the same contract.                                                                                      |
| method   | <code>InWrapX...(self, value: int) -&gt; int</code>                                              | Wrap an x coordinate when x is periodic; otherwise return it unchanged; corresponding Y/Z variants use the same contract.                                                               |

## PopGrid

| Kind     | Declaration                                                                                      | Description                                                                                                                                                                             |
|----------|--------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| method   | <code>__len__(self) -&gt; int</code>                                                             | Return the number of lattice sites.                                                                                                                                                     |
| property | <code>xDim...(self) -&gt; int</code>                                                             | Number of lattice sites along x; corresponding Y/Z variants use the same contract.                                                                                                      |
| property | <code>nDims(self) -&gt; int</code>                                                               | Number of spatial dimensions.                                                                                                                                                           |
| property | <code>wrapX...(self) -&gt; bool</code>                                                           | Whether x uses periodic boundary wrapping; corresponding Y/Z variants use the same contract.                                                                                            |
| method   | <code>Box(self, x1: int, x2: int, y1: int=None, y2: int=None, z1: int=None, z2: int=None)</code> | Iterate lattice coordinates in the half-open rectangular region $[x1, x2) \times [y1, y2) \times [z1, z2)$ ; omit unused axes in lower dimensions.                                      |
| method   | <code>Hood(self, hood, x: int, y: int=None, z: int=None, *, unroll: bool=False)</code>           | Apply relative neighborhood offsets at a lattice coordinate, respecting wrapped and nonwrapped boundaries.<br><code>unroll=True</code> enables compiled-loop expansion where supported. |
| method   | <code>ToI(self, x: int, y: int=-1, z: int=-1) -&gt; int</code>                                   | Convert lattice coordinates to a linear site index.                                                                                                                                     |
| method   | <code>ItoX...(self, i: int) -&gt; int</code>                                                     | Convert a linear site index to its x coordinate; corresponding Y/Z variants use the same contract.                                                                                      |

| Kind   | Declaration                                                      | Description                                                                                                                 |
|--------|------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| method | <code>InWrapX...(self, value: int) -&gt; int</code>              | Wrap an x coordinate when x is periodic; otherwise return it unchanged; corresponding Y/Z variants use the same contract.   |
| method | <code>__getitem__(self, key) -&gt; int</code>                    | Return the integer population count at the selected site.                                                                   |
| method | <code>__setitem__(self, key, value: int)</code>                  | Set the population count at the selected site immediately; this is distinct from queueing a delta with <code>Add()</code> . |
| method | <code>Add(self, value: int, x: int, y: int=-1, z: int=-1)</code> | Queue a signed population-count delta at a site; call <code>Update()</code> to apply pending deltas.                        |
| method | <code>Update(self)</code>                                        | Apply queued population-count deltas to the current counts.                                                                 |
| method | <code>Reset(self)</code>                                         | Clear population counts and any pending deltas.                                                                             |
| method | <code>GetPop(self) -&gt; int</code>                              | Return the total population summed across all lattice sites.                                                                |
| method | <code>All(self) -&gt; np.ndarray</code>                          | Return linear indices of sites whose population count is nonzero.                                                           |

## PDEgrid

| Kind     | Declaration                                                                                      | Description                                                                                                                                                                          |
|----------|--------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| method   | <code>__len__(self) -&gt; int</code>                                                             | Return the number of lattice sites.                                                                                                                                                  |
| property | <code>xDim...(self) -&gt; int</code>                                                             | Number of lattice sites along x; corresponding Y/Z variants use the same contract.                                                                                                   |
| property | <code>nDims(self) -&gt; int</code>                                                               | Number of spatial dimensions.                                                                                                                                                        |
| property | <code>wrapX...(self) -&gt; bool</code>                                                           | Whether x uses periodic boundary wrapping; corresponding Y/Z variants use the same contract.                                                                                         |
| method   | <code>Box(self, x1: int, x2: int, y1: int=None, y2: int=None, z1: int=None, z2: int=None)</code> | Iterate lattice coordinates in the half-open rectangular region $[x1, x2) \times [y1, y2) \times [z1, z2)$ ; omit unused axes in lower dimensions.                                   |
| method   | <code>Hood(self, hood, x: int, y: int=None, z: int=None, *, unroll: bool=False)</code>           | Apply relative neighborhood offsets at a lattice coordinate, respecting wrapped and nonwrapped boundaries. <code>unroll=True</code> enables compiled-loop expansion where supported. |
| method   | <code>ToI(self, x: int, y: int=-1, z: int=-1) -&gt; int</code>                                   | Convert lattice coordinates to a linear site index.                                                                                                                                  |
| method   | <code>ItoX...(self, i: int) -&gt; int</code>                                                     | Convert a linear site index to its x coordinate; corresponding Y/Z variants use                                                                                                      |

| Kind   | Declaration                                                                                                                                                                    | Description                                                                                                                                          |
|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
|        |                                                                                                                                                                                | the same contract.                                                                                                                                   |
| method | <code>InWrapX...(self, value: int) -&gt; int</code>                                                                                                                            | Wrap an x coordinate when x is periodic; otherwise return it unchanged; corresponding Y/Z variants use the same contract.                            |
| method | <code>__getitem__(self, key) -&gt; float</code>                                                                                                                                | Return the field value at the selected site.                                                                                                         |
| method | <code>__setitem__(self, key, value: float)</code>                                                                                                                              | Set the field value at the selected site immediately; this is distinct from queueing a change with <code>Add()</code> .                              |
| method | <code>Add(self, value: float, x: int, y: int=-1, z: int=-1)</code>                                                                                                             | Queue a field-value delta at a site; call <code>Update()</code> to apply pending changes.                                                            |
| method | <code>Update(self)</code>                                                                                                                                                      | Apply queued field changes to the current field.                                                                                                     |
| method | <code>Reset(self)</code>                                                                                                                                                       | Clear the field and any pending changes.                                                                                                             |
| method | <code>SetTimeSpaceStep(self, dt: float, dx: float, dy: float=1.0, dz: float=1.0)</code>                                                                                        | Set the temporal step <code>dt</code> and spatial steps <code>dx</code> , <code>dy</code> , and <code>dz</code> used by numerical transport methods. |
| method | <code>Dx...(self) -&gt; float</code>                                                                                                                                           | Return the configured spatial step along x; corresponding Y/Z variants use the same contract.                                                        |
| method | <code>Dt(self) -&gt; float</code>                                                                                                                                              | Return the configured time step.                                                                                                                     |
| method | <code>Diffusion(self, rateConstant: float, xMinBC=None, xMaxBC=None, yMinBC=None, yMaxBC=None, zMinBC=None, zMaxBC=None)</code>                                                | Apply scalar-coefficient diffusion using the configured grid spacing and optional per-face boundary conditions.                                      |
| method | <code>DiffusionMask(self, rateConstant: float, mask=None, xMinBC=None, xMaxBC=None, yMinBC=None, yMaxBC=None, zMinBC=None, zMaxBC=None)</code>                                 | Diffuse across unmasked interfaces, with optional boundary values on active domain faces; masked cells remain impermeable.                           |
| method | <code>DiffusionField(self, rateConstants, xMinBC=None, xMaxBC=None, yMinBC=None, yMaxBC=None, zMinBC=None, zMaxBC=None)</code>                                                 | Apply diffusion with spatially varying rate constants and optional boundary conditions.                                                              |
| method | <code>DiffusionInterfaces(self, rateConstantsX, rateConstantsY=None, rateConstantsZ=None, xMinBC=None, xMaxBC=None, yMinBC=None, yMaxBC=None, zMinBC=None, zMaxBC=None)</code> | Apply diffusion using rate constants defined on grid interfaces; separate arrays may specify x, y, and z interfaces.                                 |
| method | <code>DiffusionADI(self, rateConstant: float, xMinBC=None, xMaxBC=None, yMinBC=None, yMaxBC=None, zMinBC=None, zMaxBC=None)</code>                                             | Apply diffusion using an alternating-direction implicit method.                                                                                      |

| Kind   | Declaration                                                                                                                                         | Description                                                                                 |
|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------|
| method | <code>DiffusionRadialCircle(self, rateConstant: float, outerBC=None)</code>                                                                         | Apply radial diffusion on a circular domain with an optional outer boundary condition.      |
| method | <code>DiffusionRadialSphere(self, rateConstant: float, outerBC=None)</code>                                                                         | Apply radial diffusion on a spherical domain with an optional outer boundary condition.     |
| method | <code>Advection(self, vx: float, vy: float=0.0, vz: float=0.0, xMinBC=None, xMaxBC=None, yMinBC=None, yMaxBC=None, zMinBC=None, zMaxBC=None)</code> | Apply advection using a constant velocity vector and optional per-face boundary conditions. |
| method | <code>AdvectionField(self, xVels, yVels=None, zVels=None, xMinBC=None, xMaxBC=None, yMinBC=None, yMaxBC=None, zMinBC=None, zMaxBC=None)</code>      | Apply advection using spatially varying velocity fields.                                    |
| method | <code>AdvectionInterfaces(self, xVels, yVels=None, zVels=None, xMinBC=None, xMaxBC=None, yMinBC=None, yMaxBC=None, zMinBC=None, zMaxBC=None)</code> | Apply advection using velocity values defined on grid interfaces.                           |

## PixWindow visualization

```
StartPixWindow(xDim, yDim, scale=1, title='PAL', headless=False) -> tuple[Pix, PixWindow]
```

Create a pixel-based visualization window and its writable pixel buffer.

| Kind   | Declaration                                       | Description                                                                                        |
|--------|---------------------------------------------------|----------------------------------------------------------------------------------------------------|
| method | <code>IsOpen(self) -&gt; bool</code>              | Return whether the window is currently open.                                                       |
| method | <code>Update(self)</code>                         | Publish the current pixel buffer as a frame.                                                       |
| method | <code>Save(self, path: str, block=False)</code>   | Save the current pixel buffer to an image file; block controls whether to wait for completion.     |
| method | <code>StartGif(self, path: str, delay=100)</code> | Start recording frames to a GIF at path with the requested frame delay in milliseconds.            |
| method | <code>AddGifFrame(self, block=False)</code>       | Append the current pixel buffer as a GIF frame; optionally wait for the frame operation to finish. |
| method | <code>StopGif(self)</code>                        | Finish GIF recording and write the accumulated animation.                                          |
| method | <code>Close(self)</code>                          | Close the window and release its rendering resources.                                              |

## OpenGLWindow visualization

```
StartOpenGLWindow(xDim, yDim, zDim=None, width=800, height=800, title='PAL', headless=False) ->
```

```
tuple[OpenGLDraw, OpenGLWindow]
```

Create an OpenGL visualization window for 2D or 3D drawing.

| Kind   | Declaration                                              | Description                                                                                         |
|--------|----------------------------------------------------------|-----------------------------------------------------------------------------------------------------|
| method | <code>Update(self)</code>                                | Publish the accumulated drawing primitives as the next frame.                                       |
| method | <code>IsOpen(self) -&gt; bool</code>                     | Return whether the window is currently open.                                                        |
| method | <code>StartGif(self, path, delay=100)</code>             | Start recording rendered frames to a GIF with the requested frame delay in milliseconds.            |
| method | <code>AddGifFrame(self, block=False, timeout=30)</code>  | Capture the current scene as a GIF frame; optionally wait for completion.                           |
| method | <code>StopGif(self, timeout=30)</code>                   | Finish GIF recording and write the animation.                                                       |
| method | <code>Borders(self, width, color)</code>                 | Draw persistent outline borders around the model domain.                                            |
| method | <code>Camera(self, x, y, z, yaw=None, pitch=None)</code> | Set the 3D camera position and optional yaw/pitch angles.                                           |
| method | <code>Clear(self)</code>                                 | Clear accumulated drawing primitives for the next frame.                                            |
| method | <code>Background(self, color)</code>                     | Set the background color used when rendering frames.                                                |
| method | <code>Save(self, path: str, block=False)</code>          | Save the currently published scene to an image file; block controls whether to wait for completion. |
| method | <code>Close(self)</code>                                 | Close the window and release rendering resources.                                                   |

The `OpenGLWindow` table follows its source Protocol, which also contains method implementations; repeated declarations are listed once. `OpenGLDraw` is a compiled primitive collector, while the returned window handles rendering and output. Consult the [PAL Guide](#) for which object to use in Python and compiled drawing code.

## Pix drawing buffer

| Kind   | Declaration                                | Description                                                                        |
|--------|--------------------------------------------|------------------------------------------------------------------------------------|
| method | <code>__setitem__(self, key, color)</code> | Write a packed RGB color to a pixel by (x, y) coordinates or a linear pixel index. |
| method | <code>xDim(self)</code>                    | Return the pixel buffer width.                                                     |
| method | <code>yDim(self)</code>                    | Return the pixel buffer height.                                                    |
| method | <code>ToI(self, x, y)</code>               | Convert pixel coordinates (x, y) to a linear index.                                |
| method | <code>ItoX(self, i)</code>                 | Convert a linear pixel index to its x coordinate.                                  |
| method | <code>ItoY(self, i)</code>                 | Convert a linear pixel index to its y coordinate.                                  |

| Kind   | Declaration                | Description                        |
|--------|----------------------------|------------------------------------|
| method | <code>__len__(self)</code> | Return the total number of pixels. |

## OpenGLDraw primitive collector

| Kind   | Declaration                                                            | Description                                                              |
|--------|------------------------------------------------------------------------|--------------------------------------------------------------------------|
| method | <code>clear(self)</code>                                               | Discard all primitives accumulated for the next frame.                   |
| method | <code>circle(self, rad, color, x, y, z=0.0)</code>                     | Add a circle centered at (x, y, z) with radius rad and color.            |
| method | <code>Box(self, xLen, color, x, y, z=0.0, yLen=-1.0, zLen=-1.0)</code> | Add a box centered at (x, y, z); negative side lengths default to xLen.  |
| method | <code>BoxSQ(self, color, x, y, z=0.0)</code>                           | Add a unit box centered on lattice site (x, y, z).                       |
| method | <code>Line(self, width, color, x1, y1, x2, y2, z1=0.0, z2=0.0)</code>  | Add a line segment between endpoints with the requested width and color. |

## Scope and validation

This reference lists public Protocol signatures, not internal safe/fast jitclass methods. A signature does not capture every runtime overload, indexing form, or validation rule. See the PAL Guide and executable tests. Visualization constructors and window Protocol methods are included; Pix buffer and OpenGLDraw method signatures are also included; consult the PAL Guide for behavior.