

THE GenServer CHEATSHEET

Version 1.0

Initialization

```
CLIENT
def start_link(opts \\ []) do
  GenServer.start_link(__MODULE__, :ok, opts)
end
```

Returns
{:ok, pid}

```
CALLBACK
def init(:ok) do
  state = :init_state()
  {:ok, state}
end
```

```
RETURN VALUES
{:ok, state}
{ok, state, 5_000}
{:ok, state, :hibernate}
{:stop, reason*}
:ignore
```

Synchronous Operation

```
CLIENT
def sync_op(pid, args) do
  GenServer.call(pid, {:sync_op, args})
end
```

```
CALLBACK
def handle_call({:sync_op, args}, from, state) do
  new_state = f(state, args)
  {:reply, new_state}
end
```

```
RETURN VAL
{:reply, reply, new_state}
{:reply, reply, new_state, 5_000}
{:reply, reply, new_state, :hibernate}

{:noreply, new_state}
{:noreply, new_state, 5_000}
{:noreply, new_state, :hibernate}
{:stop, reason*, reply, new_state}
{:stop, reason*, new_state}
```

Asynchronous Operation

```
CLIENT
def async_op(pid, args) do
  GenServer.cast(pid, {:async_op, args})
end
```

```
CALLBACK
def handle_cast({:async_op, args}, state) do
  new_state = f(state, args)
  {:noreply, new_state}
end
```

```
RETURN VAL
{:noreply, new_state}
{:noreply, new_state, 5_000}
{:noreply, new_state, :hibernate}
{:stop, reason*, new_state}
```

THE GenServer CHEATSHEET

Version 1.0

Out of band messages

CALLBACK

```
def handle_info(msg, state) do
  new_state = f(state, msg)
  {:noreply, new_state}
end
```

RETURN VAL

```
{:noreply, new_state}
{:noreply, new_state, 5_000}
{:noreply, new_state, :hibernate}
{:stop, reason*, new_state}
```

Termination

CLIENT

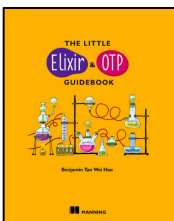
```
def stop(pid, reason \\ :normal, timeout \\ :infinity) do
  GenServer.stop(pid, reason, timeout)
end
```

CALLBACK

```
def terminate(reason, state) do
  # Perform cleanup here
  # ...
end
```

REASON*

```
:normal
:shutdown
{:shutdown, term}
term
```



The Little Elixir & OTP Guidebook

... gets you started programming applications with Elixir and OTP. You begin with a quick overview of the Elixir language syntax, along with just enough functional programming to use it effectively. Then, you'll dive straight into OTP and learn how it helps you build scalable, fault-tolerant and distributed applications through several fun examples. Come rediscover the joy of programming with Elixir and remember how it feels like to be a beginner again. *Psst! Use [tanweihao39](#) for 39% off!*