Erlang Cheatsheet

Overview of unique features

- Functional programming language
- Designed for building concurrent and distributed systems
- Uses lightweight processes instead of threads
- Supports hot-swapping of code
- Can be used for building telecom and messaging systems

Variables

```
% Declare a variable
X = 42.

% Declare a constant
- define(Y, 10).

% Declare a tuple
MyTuple = {apple, 3}.

% Declare a list
MyList = [1, 2, 3].
```

Functions

```
% Define a function
add(X, Y) ->
  X + Y.

% Call a function
Result = add(3, 4).
```

Loops

```
% Define a loop
my_loop(I) ->
  case I > 10 of
    true -> done;
    false -> my_loop(I + 1)
  end.
```

Conditionals

```erlang
% Define an if-else statement
max(X, Y) ->
  if X > Y ->
    X;
  true ->
    Y
  end.
```

Processes

```erlang
% Define a process
my_process() ->
  receive
    {From, Message} ->
      From ! {ok, Message},
      my_process()
  end.

% Start a process
Pid = spawn(my_module, my_process, []).

% Send a message to a process
Pid ! {self(), "hello"}.

% Receive a message from a process
receive
  {ok, Message} -> io:format("Received ~p", [Message])
end.
```

Resources

- Erlang documentation
- Erlang tutorial
- Erlang forum for community support and troubleshooting.