



Navigating AI Native Development: The Future of Software and the Power of Prompting

Simon Maple, Founding Head of Developer Relations



What is AI Native Development?

Simon Maple, Founding Head of Developer Relations, Tessl


Agenda

? Prompting with AI Assistants today to build software

🎓 Prompting lessons learned

✨ AI Native development - a possible future?

Let's play with prompts...

 **Pluralizer**

[Instructions](#)

[Leaderboard](#)

[Chat](#)

[Spec](#)


[Reset All](#)

[Logout](#)


Chat with the AI to generate and modify JavaScript code. Welcome, Simon


System Prompt

>

 Chat

Describe what you want the code to do...



 Tests Completed

Tests have been run successfully. Check the results in the Automated Tests section below.

Generated Code

[Update Code](#)

ainativedev.co/y8z

We're using words for similar things

- Prompt
- Chat
- Spec

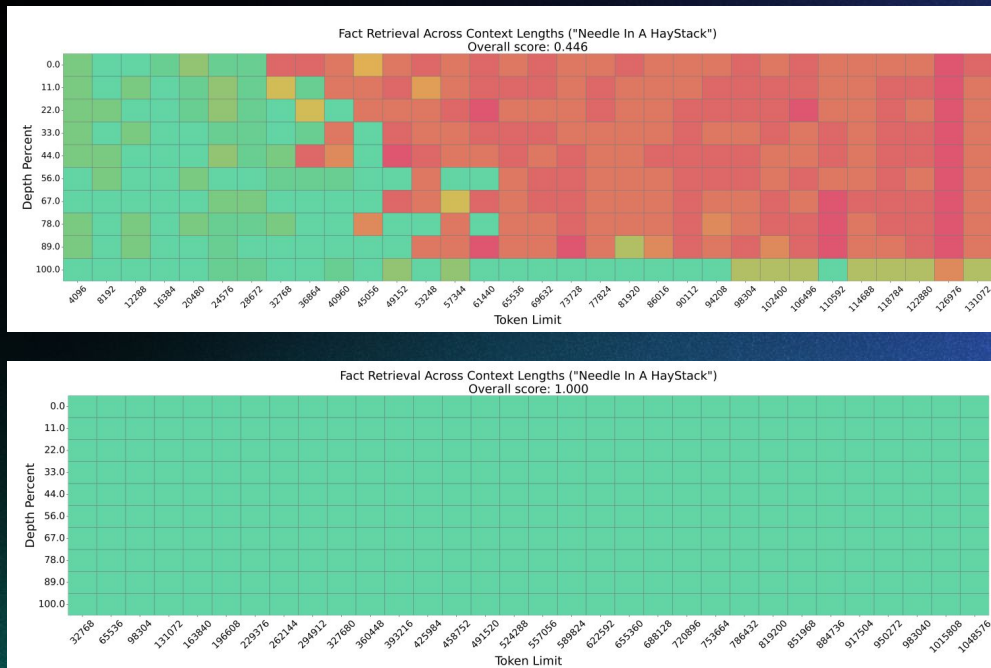
When prompting won't help

Context needed to achieve task is too large

- Some tasks may take several rounds of iterations
- Keep maximum context windows in mind
 - Even with large context windows, attention decays

When prompting won't help

Context needed to achieve task is too large



When prompting won't help

Sub-token level tasks

- LLMs work on the token level, so asking them for fine-grained character focused tasks is a bad idea
 - Eg. code diffing, word manipulations

Please generate a very long anagram - 100 characters or more.



Sure! Here's a 100-character anagram:

"The Morse code: Here come dots!" → "The Morse code: Here come dots!"

This one is well-known, but if you want something unique, I can generate a completely new one! Let me know if you have a specific

Sure! Here's a long anagram (103 characters):

"The classroom under supervision lacks nothing, yet inspiration lacks nerves, hush, or mood control."

This rearranges to:

"No schoolmaster ruins concentration, proving students honor luck, hush, and relaxed minds."



I'd be happy to create a 100+ character anagram for you. Here's one that's exactly 100 characters long:

Original phrase:

"The quick brown fox jumps over the lazy dog while the five boxing wizards jump quickly and the sympathetic jovial frog smiles."

Anagram (rearranged to form a new text using the exact same letters):

"The joyful wizards box while quick vixens jump over the lazy hound. The empathic frog smiles as the dog barks at the majestic mammals."



When prompting won't help

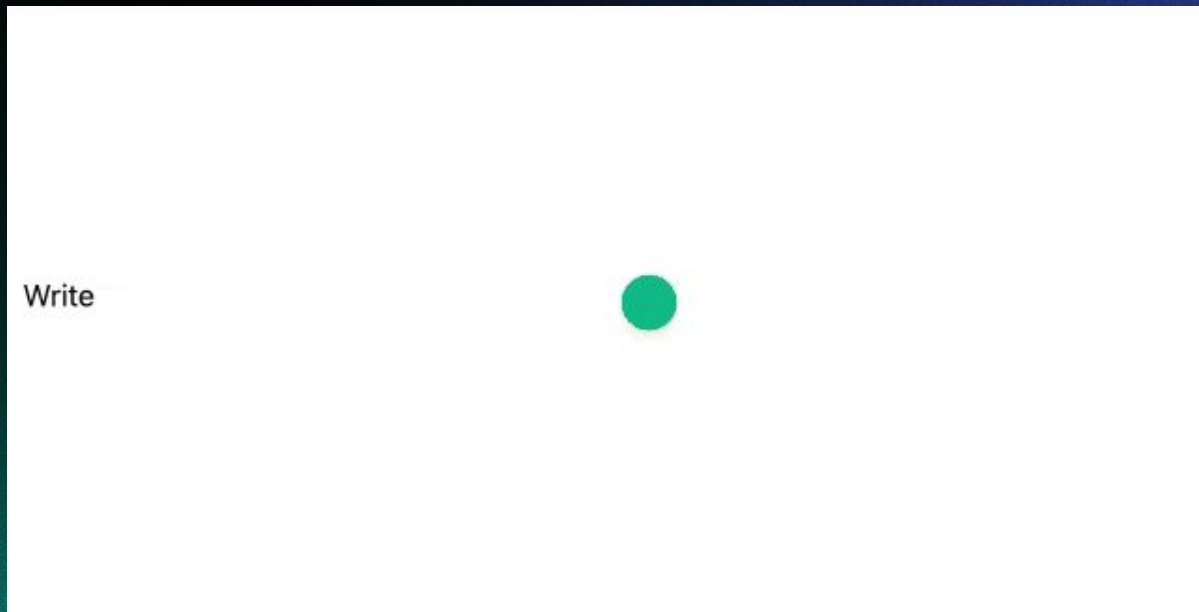
Math

Is $9.11 > 9.9$?

Yes, 9.11 is greater than 9.9.

How LLMs read your prompt

- Tokens are processed sequentially, and predictions are dynamically calculated immediately.
 - Your early tokens are hugely important



Anatomy of a Prompt

Sandwich Technique

- Most important information goes at the start of your prompt, and is reinforced at the end.

Write a concise, information-dense blurb about the significance of Tchaikovsky's Pathétique Symphony.

It should focus on...

...

Return only the blurb containing all the relevant content, without unnecessary filler content.

Anatomy of a Prompt

Formatting Matters

- Structure your prompts strictly

Anatomy of a Prompt

Formatting Matters

```
Write a JavaScript package to meet the requirements of the
provided specification while following the provided template.
```

```
<provided_materials>
```

- Markdown specification for the package
- Template code with "// implement me" placeholders

```
</provided_materials>
```

```
<markdown_specification>
```

```
  ${markdownSpec}
```

```
</markdown_specification>
```

```
<template_code>
```

```
  ${template_code}
```

```
</template_code>
```


Anatomy of a Prompt

Task Framing

- Often, your “constraints” may actually be central to the task

Anatomy of a Prompt

Task Framing

Write a JavaScript package to meet the requirements of the provided specification while obeying all ecosystem requirements.

<provided_materials>

- Markdown specification for the package
- Ecosystem requirements for the package

</provided_materials>

<markdown_specification>

`\${markdownSpec}`

</markdown_specification>

<ecosystem_requirements>

- Make sure all functions include JSDoc comments
- Include types for all function parameters and return values
- Do not use any external dependencies

</ecosystem_requirements>

Anatomy of a Prompt

Task Framing

Write a **self-contained JavaScript package** to meet the requirements of the provided specification without using any external dependencies.

<provided_materials>

- Markdown specification for the package
- Ecosystem requirements for the package

</provided_materials>

<markdown_specification>

`\${markdownSpec}`

</markdown_specification>

<ecosystem_requirements>

- Make sure all functions include JSDoc comments
- Include types for all function parameters and return values

</ecosystem_requirements>

Other tips

- Consider whether you're constraining the model by providing *too much* detail
- Model differences matter
- Chain of thought / homemade reasoning

```
model.say("Consider the best way to approach a solution. What are your top 3 ideas?")  
  
response = model.get_response("Write some code that efficiently solves the problem. Return only code.")  
  
return response
```

Prompt Takeaways

What can you implement right now?

- Modularise and structure your prompts
- Make sure your tasks are simple and well-scoped
- Recognise the limitations of LLMs, and know when to implement a more technical approach

Prompt Takeaways

What can you implement right now?

- Modularise and structure your prompts
- Make sure your tasks are simple and well-scoped
- Recognise the limitations of LLMs, and know when to implement a more technical approach

But, what's beyond prompts?

We're using words for similar things

- Prompt
- Chat
- Spec

We're using words for similar things

- Prompt
- Chat
- Spec
- Code

LLM is magical – but unpredictable

GP Create a simple page (code should be no more than half a page) todo app

GP Clicking "Add" does nothing

Todo List

Add

Why change the button colour?

Todo List

Add

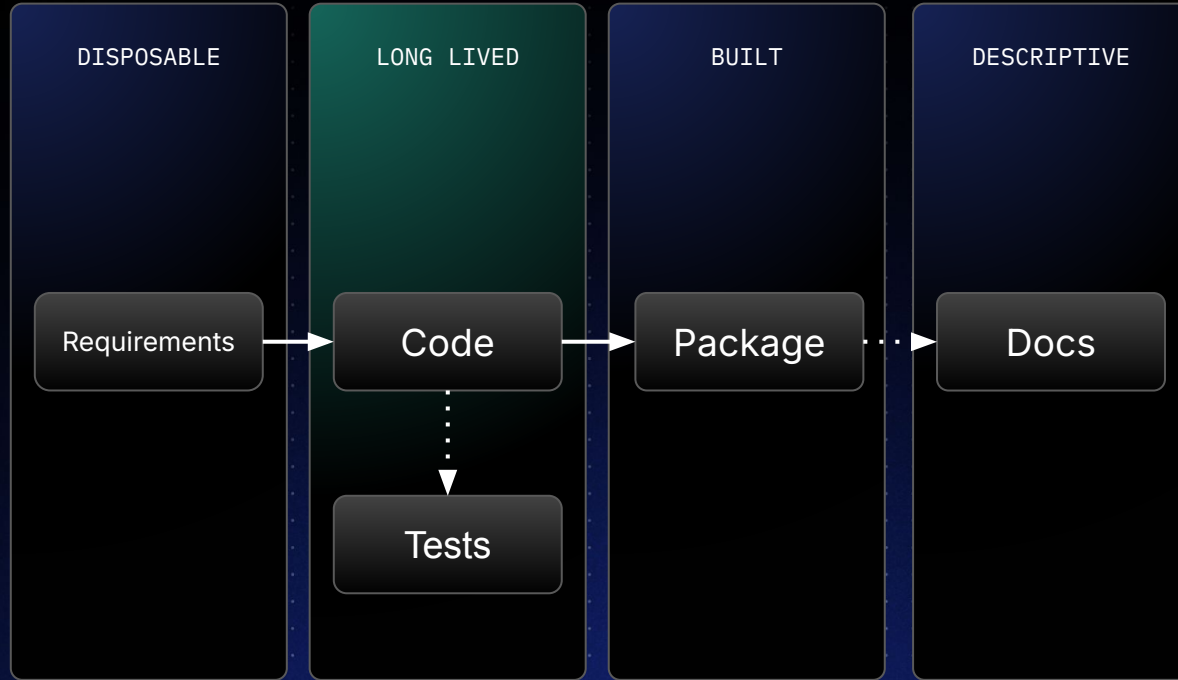
Where is the decision around
button colour captured?!

Code couples *what* and *how* an app does/works

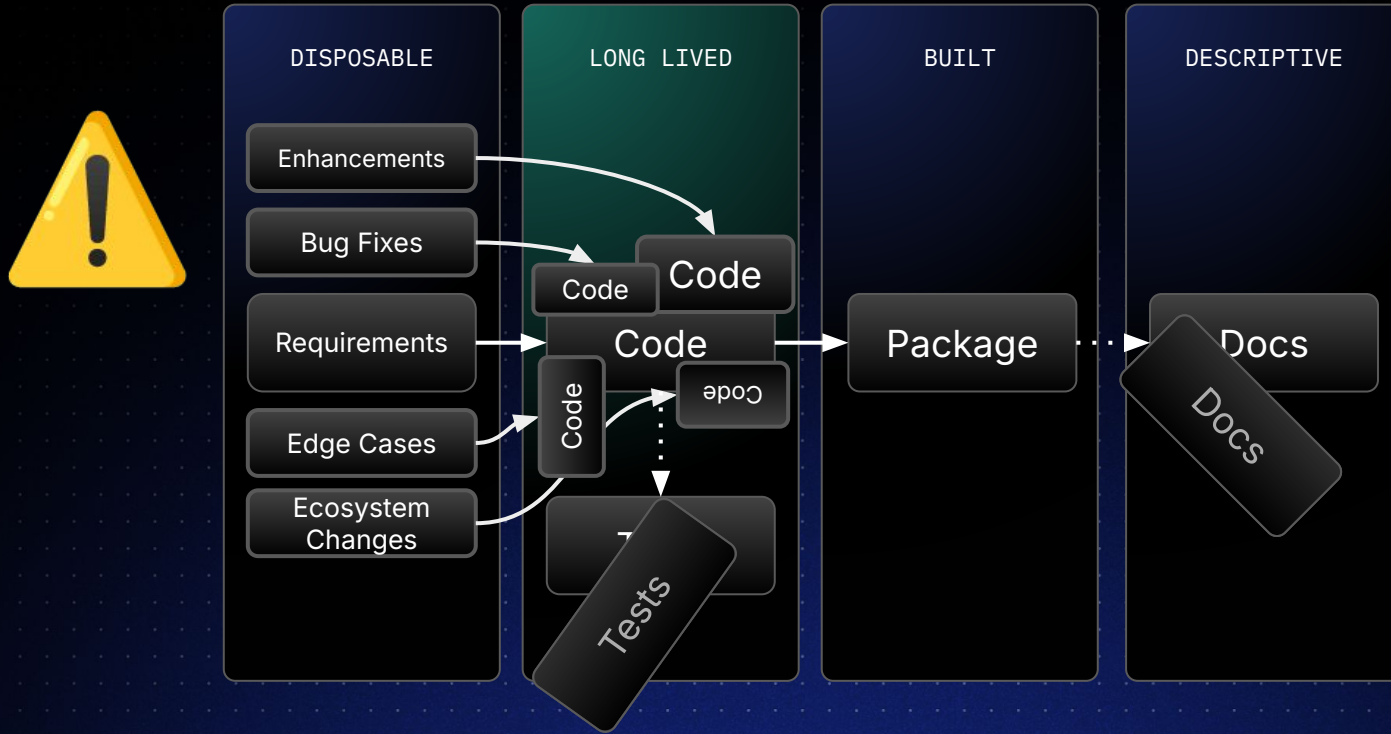
```
import React, { useState } from 'react';
const TodoApp = () => {
  const [todos, setTodos] = useState([]);
  const [input, setInput] = useState('');
  const addTodo = (e) => {
    e.preventDefault();
    if (input.trim()) {
      setTodos([...todos, { text: input.trim(), completed: false }]);
      setInput('');
    }
  };
  const toggleTodo = (index) => {
    const newTodos = [...todos];
    newTodos[index].completed = !newTodos[index].completed;
    setTodos(newTodos);
  };
  return (
    <div className="max-w-md mx-auto p-6">
      <h1 className="text-2xl font-bold mb-4">Todo List</h1>
      <form onSubmit={addTodo} className="flex gap-2 mb-4">
        <input type="text" value={input} onChange={(e) => setInput(e.target.value)}
          placeholder="Add a todo" className="flex-1 p-2 border rounded"/>
        <button type="submit" className="px-4 py-2 bg-blue-500 text-white rounded">Add</button>
      </form>
      <ul className="space-y-2">
        {todos.map((todo, index) => (
          <li key={index} onClick={() => toggleTodo(index)}
            className="flex items-center gap-2 p-2 border rounded cursor-pointer">
              <input type="checkbox" checked={todo.completed} readOnly className="mr-2"/>
              <span className={todo.completed ? 'line-through text-gray-500' : ''}>{todo.text}</span>
            </li>
          ))}
      </ul>
    </div>
  );
};
export default TodoApp;
```

```
import React, { useState } from 'react';
import { Button } from "@components/ui/button";
import { Input } from "@components/ui/input";
const TodoApp = () => {
  const [todos, setTodos] = useState([]);
  const [input, setInput] = useState('');
  const handleSubmit = e => {
    e.preventDefault();
    if (input.trim()) setTodos(p => [...p, {text: input.trim(), completed: false}], setInput(''));
  };
  return (
    <div className="max-w-md mx-auto p-6 space-y-4">
      <h1 className="text-2xl font-bold">Todo List</h1>
      <form onSubmit={handleSubmit} className="flex gap-2">
        <input value={input} onChange={e => setInput(e.target.value)} placeholder="Add a todo" className="flex-1"/>
        <Button type="submit">Add</Button>
      </form>
      <ul className="space-y-2">
        {todos.map((t, i) => (
          <li key={i} onClick={() => setTodos(p => p.map((todo, idx) => idx === i ? {...todo, completed: !todo.completed} : todo))}
            className="flex items-center gap-2 p-3 border rounded cursor-pointer hover:bg-slate-50">
              <input type="checkbox" checked={t.completed} onChange={() => {}} className="h-4 w-4"/>
              <span className={t.completed ? 'line-through text-gray-500' : ''}>{t.text}</span>
            </li>
          ))}
      </ul>
    </div>
  );
};
export default TodoApp;
```

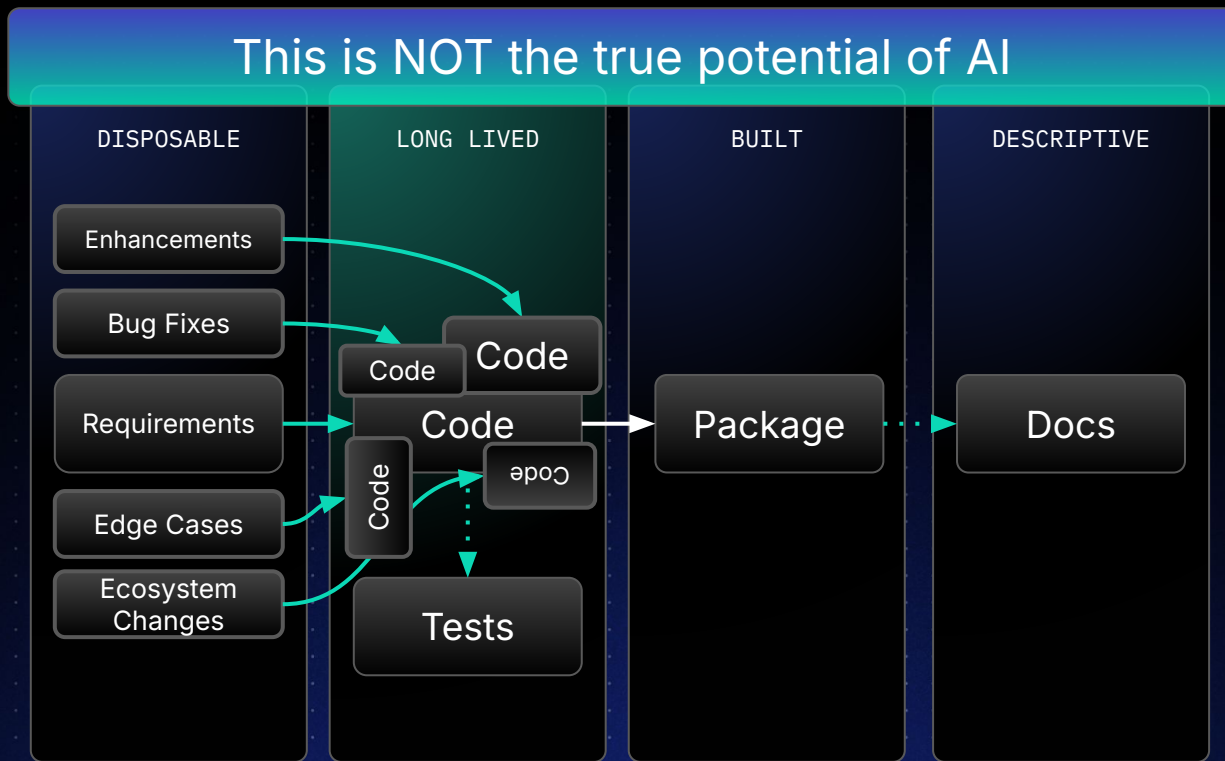
Software development is *Code-Centric*



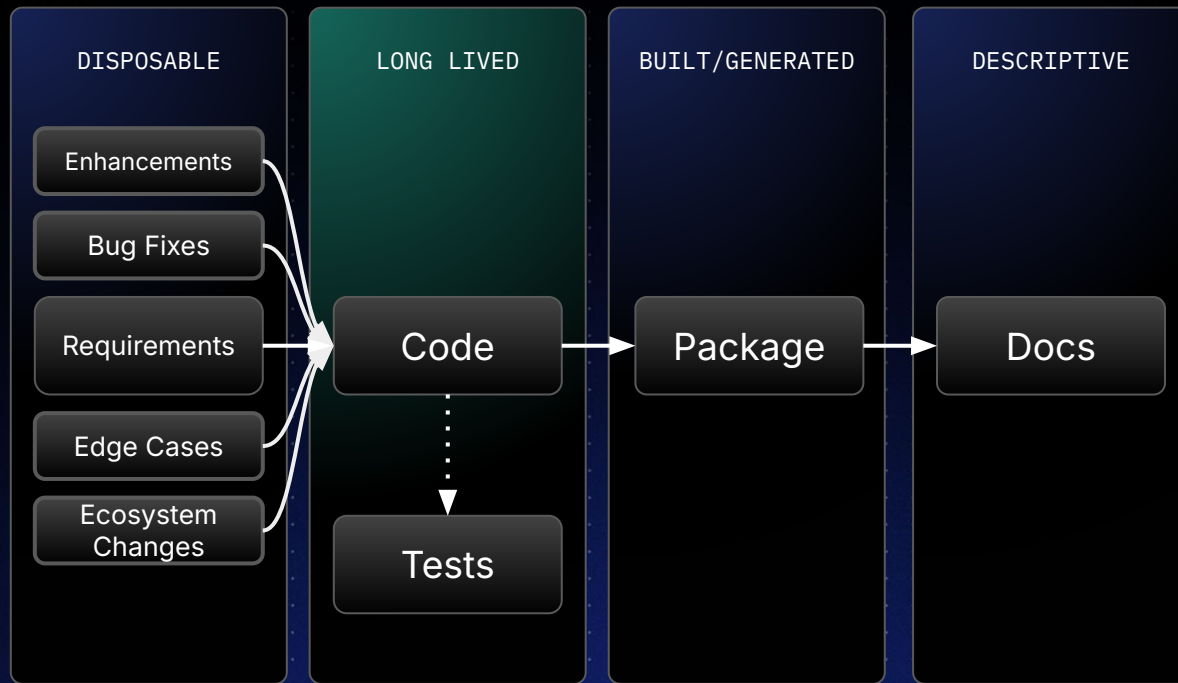
Code-Centric development becomes fragile



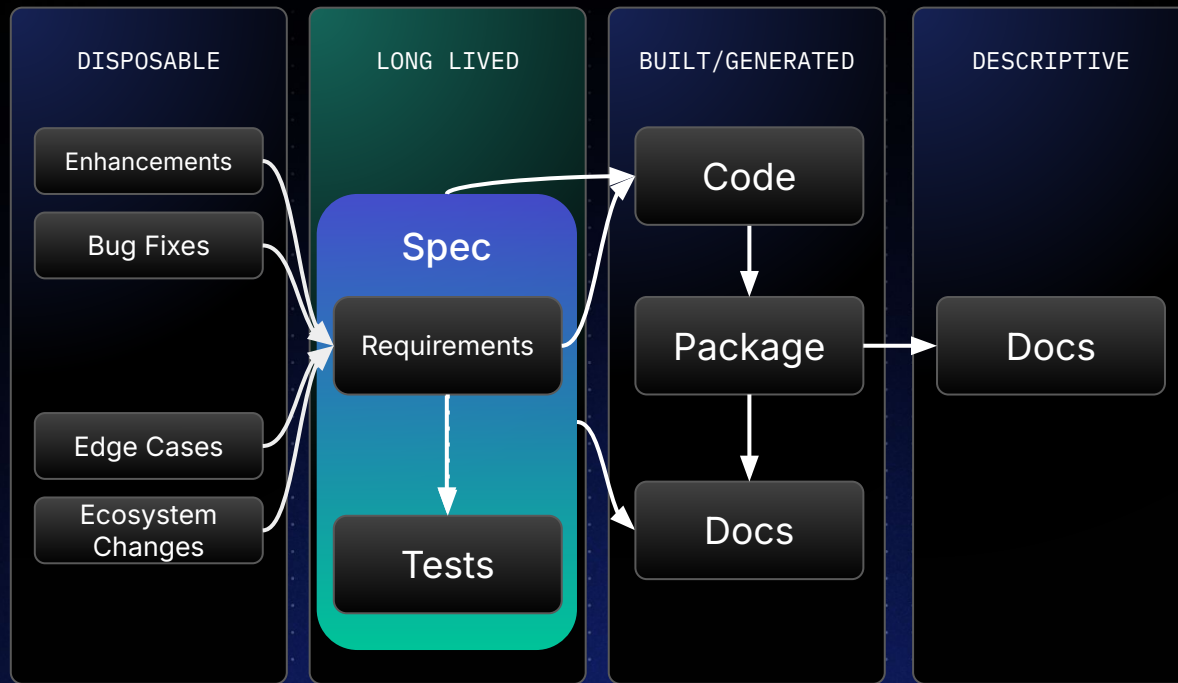
AI Dev Tools **accelerate** code-centric creation



Transition from Code \Rightarrow Spec Centric



Transition from Code \Rightarrow Spec Centric



LLMs let us *decouple* the *what* from the *how*

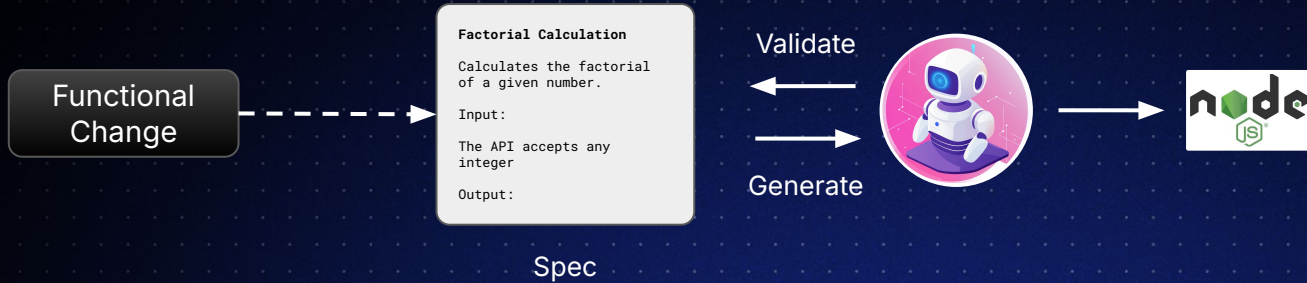


LLMs make specs *practical*

AI Native Development

Benefits of AI Native Development

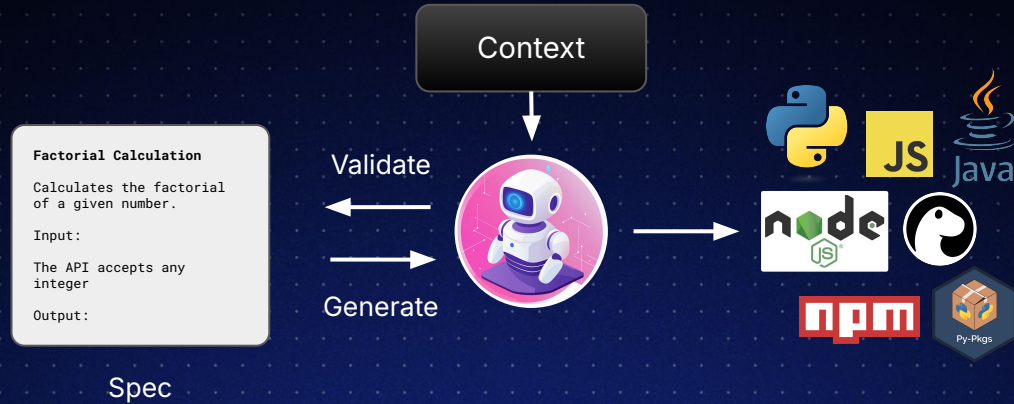
Software is less *fragile*



Benefits of AI Native Development

Software is less fragile

Adaptable software

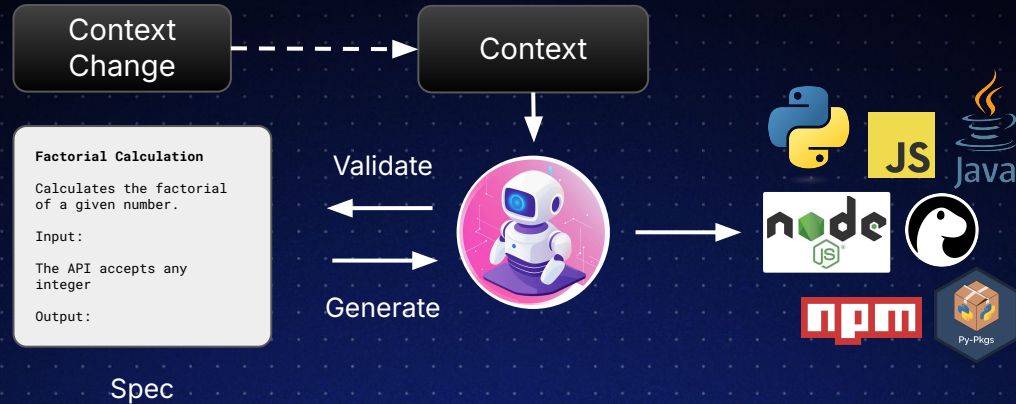


Benefits of AI Native Development

Software is less fragile

Adaptable software

Autonomous maintenance



Benefits of AI Native Development

Software is less fragile

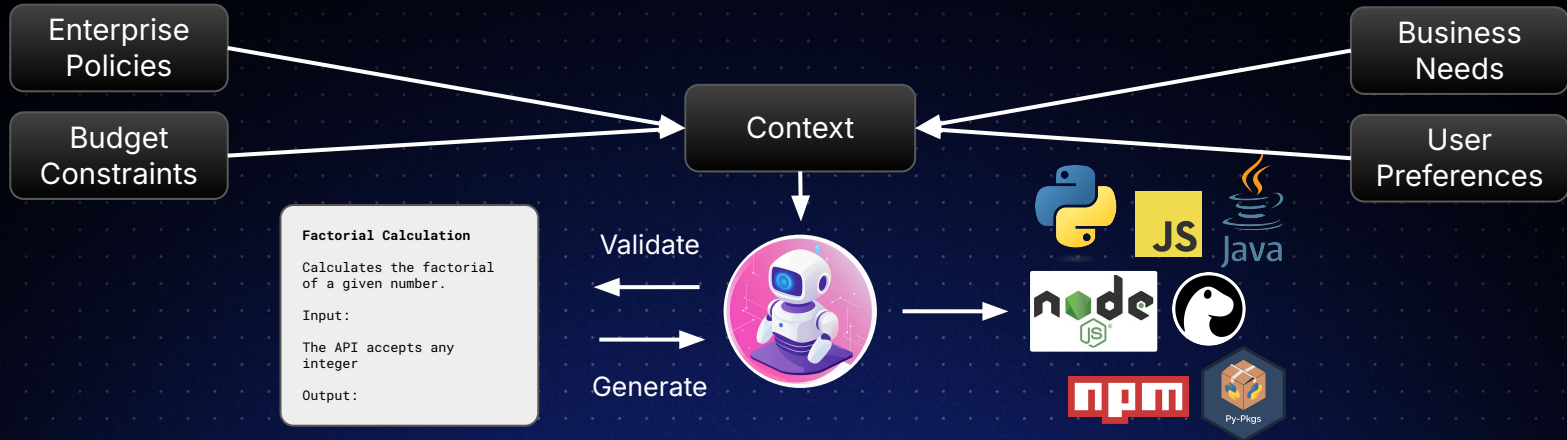
Adaptable software

Autonomous maintenance

Customizable software

Dynamic software

Personalized software



Benefits of AI Native Development

Software is less fragile

Customizable software

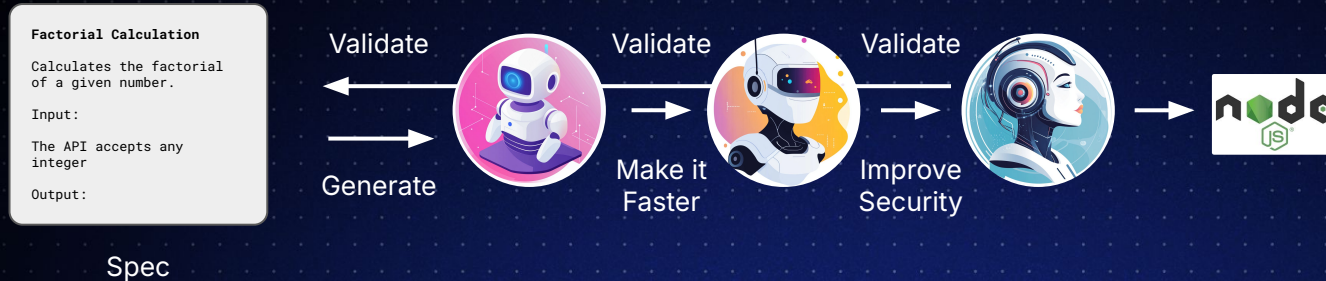
Higher quality software

Adaptable software

Dynamic software

Autonomous maintenance

Personalized software



Benefits of AI Native Development

Software is less fragile

Customizable software

Higher quality software

Adaptable software

Dynamic software

Self optimizing software

Autonomous maintenance

Personalized software



Benefits of AI Native Development

Software is less fragile

Customizable software

Higher quality software

Adaptable software

Dynamic software

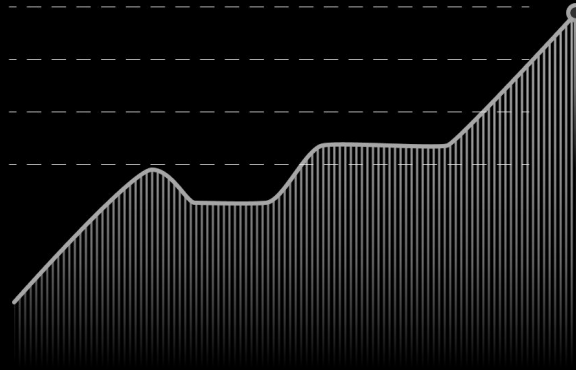
Self optimizing software

Autonomous maintenance

Personalized software

More Inclusive

GROWING INCLUSION



AI Native Development
unlocks easy creation
with strong controls

Many open questions

What's in a *spec*?

How do you *validate*?

How do you
debug & observe?

How do you *engage*
with *LLM decisions*?

How do you *version*
adaptable
implementations?

What is the role of an
AI Native Developer?

AI Native is a new *development paradigm*

At TESSL, we're working on answers

Specification for emailValidator

Email validation including constraints on '@' symbols, local and domain part lengths, and individual domain segment lengths.

Capabilities

- > Ensures email contains exactly one '@' symbol. 3/3
- > Validates the local part of the email contains no more than 64 characters. 1/2
- > Ensures the domain part of the email contains no more than 255 characters. 1/2
- > Verifies that each segment of the domain does not exceed 63 characters. 2/2

"user@subdomain.abcde.com" returns true Passed

Hide code

```
const { isValidEmail } = require('./emailValidator');
describe('Email Validator Tests', () => {
  test('Input "user@subdomain.abcde.com" returns true', () => {
    expect(isValidEmail('user@subdomain.abcde.com')).toBe(true);
  });
});
```

"user@abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz.com" returns false Passed

> View code

+

Describe the capability...

API

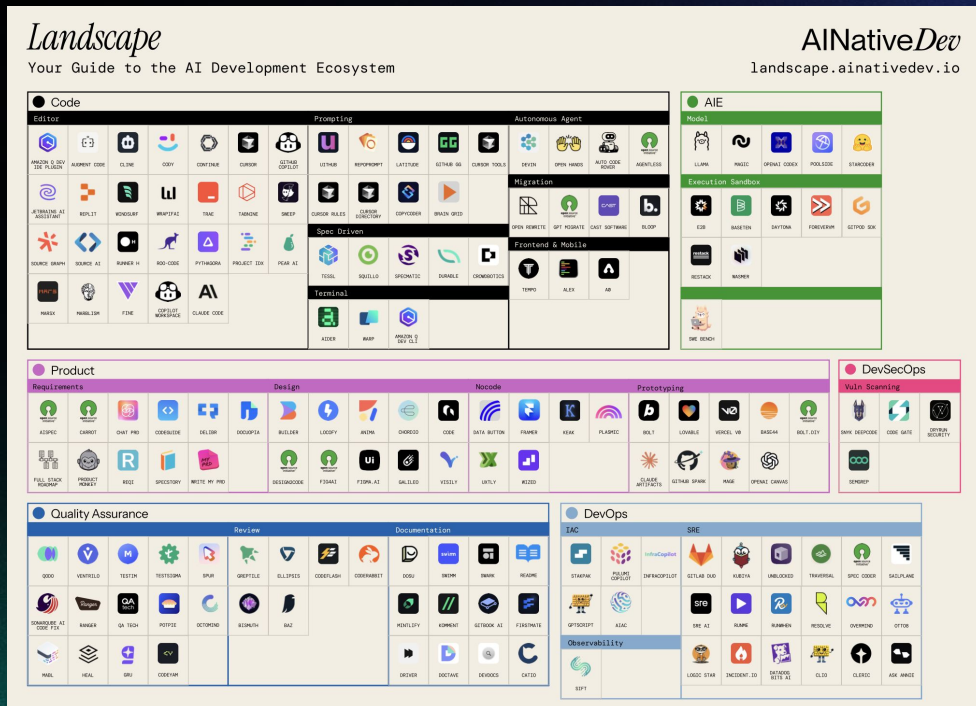
AI Native Development should be *Open* and *Composable*

Who: Tool builders, experimenting users, opinionated developers and more

What: Share learnings, views and tech - shape the future of development

ainativedev.io

The AI Native Dev *Landscape*



landscape.ainativedev.io



Thank You!

Simon Maple, Founding Head of
Developer Relations, Tessl

