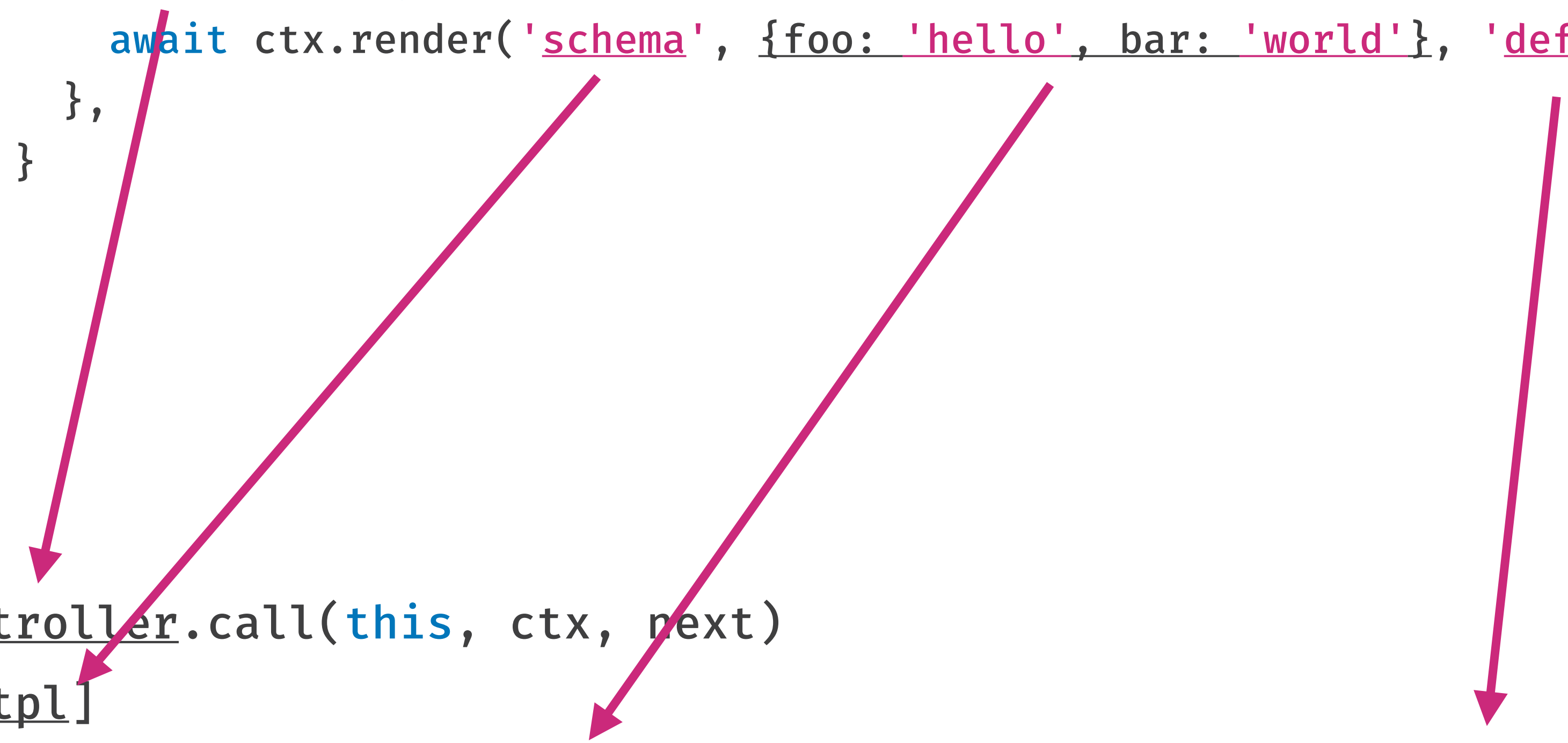


Controller Schema 解析过程

- ✓ routes register
- ✓ init ctx.render() & run

```
export default {  
  urls: ['/schema', '/foo'],  
  middlewares: ['schema-logger'],  
  controller: async (ctx) => {  
    await ctx.render('schema', {foo: 'hello', bar: 'world'}, 'default')  
  },  
}
```

```
app.use( async ctx => {  
  const serveData = await controller.call(this, ctx, next)  
  const Template = ctx.$tpls[tpl]  
  const template = new Template({ ctx, page, serveData, serveBundle: ctx.$bundles[page] })  
  ctx.body = await template.toHtml()  
})
```



Template + pageInitData + bundle.js = HTML

HTML Shell

```
<!DOCTYPE html>
<html>
  <head>
    <script type="text/javascript">
      window.serveData = {pageInitData}
    </script>
  </head>
  <body>
    <div id="app">{pageInitData + bundle.js → VueSSRString}</div>
  </body>
</html>
```

The diagram illustrates the assembly of an HTML document. At the top, the equation 'Template + pageInitData + bundle.js = HTML' is shown. Below it, a red dashed box represents the 'HTML Shell'. A red arrow points from 'Template' to the opening <html> tag. A green arrow points from 'pageInitData' to the script block in the head and the content inside the app div. A blue arrow points from 'bundle.js' to the script block in the head and the content inside the app div. The HTML code inside the shell shows the structure: <!DOCTYPE html>, <html>, <head>, <script type="text/javascript">, window.serveData = {pageInitData}, </script>, </head>, <body>, <div id="app">{pageInitData + bundle.js → VueSSRString}</div>, </body>, and </html>.