#自动引入Api

使用 unplugin-auto-import (opens new window)可以自动导入api，不需要import。
没有使用时

```
import { ref, computed } from 'vue'
const count = ref(0)
const doubled = computed(() => count.value * 2)

```

使用插件后

```
const count = ref(0)
const doubled = computed(() => count.value * 2)

```

#安装扩展

首先安装扩展包

```
yarn add -D unplugin-auto-import

```

#vite.config.ts

下面是修改vite.config.js配置文件，来增加对vue 与 vue-router的 api自动引用

```
import { defineConfig } from 'vite'
import vue from '@vitejs/plugin-vue'

import AutoImport from 'unplugin-auto-import/vite'

export default defineConfig({
  plugins: [
    vue(),
    AutoImport({
```

```
      imports: ['vue', 'vue-router'],
      //为true时在项目根目录自动创建
      dts: 'types/auto-imports.d.ts',
    }),
  ]
})

```

#tsconfig

接下来在 tsconfig.json中引入生成的类型声明文件，当执行 yarn dev 时会根据上面定义的 dts选项自动生成类型声明文件 types/auto-imports.d.ts

```
"include": [
  ...
  "types/**/*.ts"
]

```

#使用示例

现在在vue组件与.ts文件中使用ref等vue的api，就不需要import了

```
const user = ref('houdunren.com@向军大叔')

```

#element plus

Element plus与ant design也提示了针对该插件的支持，配置好后也不需要import api了。
下面是element plus的vite.config.ts的配置项

```
import { Plugin } from 'vite'
import AutoImport from 'unplugin-auto-import/vite'
import Components from 'unplugin-vue-components/vite'
import { ElementPlusResolver } from 'unplugin-vue-components/resolvers'

export default function (plugins: Plugin[]) {
  plugins.push(
```

```
    AutoImport({
        //引入element plus自动api支持
      resolvers: [ElementPlusResolver()],
      imports: ['vue', 'vue-router'],
      //为true时在项目根目录自动创建
      dts: 'types/auto-imports.d.ts',
    })
  )
}

```

现在我们可以直接使用 element plus 的api，而不需要import进来再使用了

```
ElMessage.success('houdunren.com')

```

不过有些依赖的css会识别不到造成样式错误，建议在index.html 项目模板中引入样式

```
<link rel="stylesheet" href="//unpkg.com/element-plus/dist/index.css" />

```

#自动加载组件

使用 unplugin-vue-components (opens new window)可以自动按需要加载组件，我们常用的 ant design 、element plus已经基于该插件实现了按需加载。

#安装配置

下面我们通过配置 unplugin-vue-components (opens new window)插件来实现自动 import 组件，节省我们import的代码

#安装插件

首先安装需要的扩展包

```
yarn add -D unplugin-vue-components

```

#vite.config.ts

下面在vite中配置组件的按需自动加载

```
import { defineConfig } from 'vite'
import vue from '@vitejs/plugin-vue'

import Components from 'unplugin-vue-components/vite'

export default defineConfig({
  plugins: [
    vue(),
    Components({
        //自动加载的组件目录，默认值为 ['src/components']
      dirs: ['src/components'],
      //组件名称包含目录，防止同名组件冲突
      directoryAsNamespace: true,
      //指定类型声明文件，为true时在项目根目录创建
      dts: 'types/components.d.ts',
    })
  ]
})

```

#tsconfig.json

接下来在 tsconfig.json中引入生成的类型声明文件，当执行 yarn dev 时会根据上面定义的 dts选项自动生成类型声明文件 types/components.d.ts

```
"include": [
  ...
  "types/**/*.ts"
]

```

#组件重名

插件的默认配置会加载components目录中的组件，所以当存在相同名称的组件user.vue时，会报下面的错误

```

```

```
[unplugin-vue-components] component "User"(xxxxx/user.vue) has naming conflicts with other components, ignored.

```

我们可以声明组件的注册名称包含目录，这样组件包含目录前缀来避免组件重名

```
Components({
  resolvers: [ElementPlusResolver()],
  directoryAsNamespace: true,
  dts: true,
})

```

#使用示例

运行 yarn dev 命令后，我们现在可以在组件中直接使用 src/components中的组件，而不需要import引入组件了。

因为定义了 `directoryAsNamespace` 选项，所以实际使用的组件名称前要加上目录名，vscode也会自动根据TS类型进行提示的

```
<WangEditorEditor/>

```

#element plus

Element plus与ant design也提示了针对该插件的支持，配置好后也不需要import 组件了。

下面是element plus的vite.config.ts的配置项

```
import { Plugin } from 'vite'
import AutoImport from 'unplugin-auto-import/vite'
import Components from 'unplugin-vue-components/vite'
import { ElementPlusResolver } from 'unplugin-vue-components/resolvers'

export default function (plugins: Plugin[]) {
  plugins.push(
    Components({
      dirs: ['src/components'],
      directoryAsNamespace: true,
      //为true时在项目根目录自动创建
```

```
      dts: 'types/components.d.ts',
      //引入element plus自动组件支持
      resolvers: [ElementPlusResolver()],
    })
  )
}

```

现在我们可以直接使用 element plus 的组件了，而不需要import进来再使用了

```
<el-button type="primary" size="default">houdunren.com</el-button>

```

有些依赖的css会识别不到造成样式错误，建议在index.html 项目模板中引入样式

```
<link rel="stylesheet" href="//unpkg.com/element-plus/dist/index.css" />

```