GMTC
全球大前端技术大会

Vue 中的 TypeScript 支持

某次大型活动中的 TypeScript 实践

赵缙翔

Beijing, C hina
2023

赵缙翔

快手前端工程师

@vue/composition-api, unpl ugi n-vue2-scri pt-setup, volar 等项目的 Colla borator

xiaoxiangm

知oe

某兔

目录

我以为的活动页


雅思补习班
优惠限时抢
¥500
立即领取
课程简介


我的2018
时空轨迹


双11狂欢季
美妆护肤全场大促
10元
满199使用
立即领取
30元
满299使用
立即领取
50元
满399使用
立即领取


幸运大转盘
红包转出来
微盘钱包优势
福利多

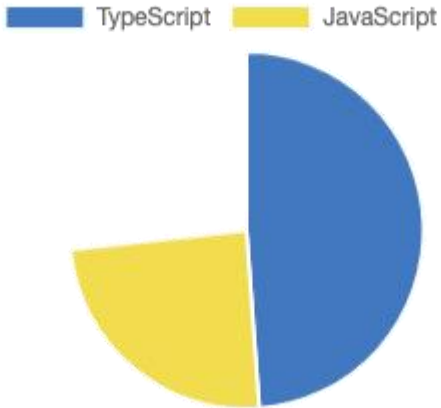
我们的CNY活动页

活动规模

2022 年春节项目

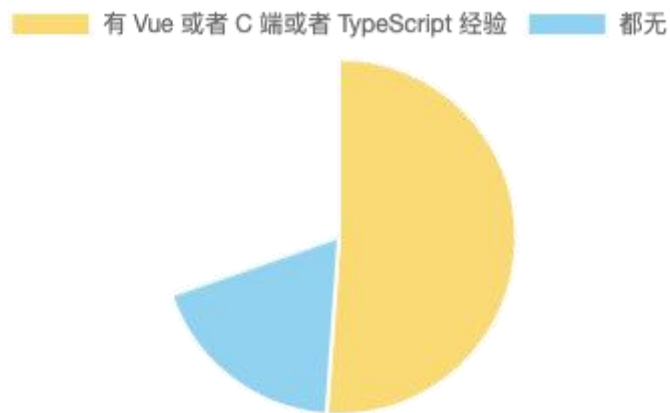
会场数量	13
开发人数	50
提交数量	7000
文件数量	3000
代码数量	25万行

开发者技术能力调研


TypeScript
JavaScript only

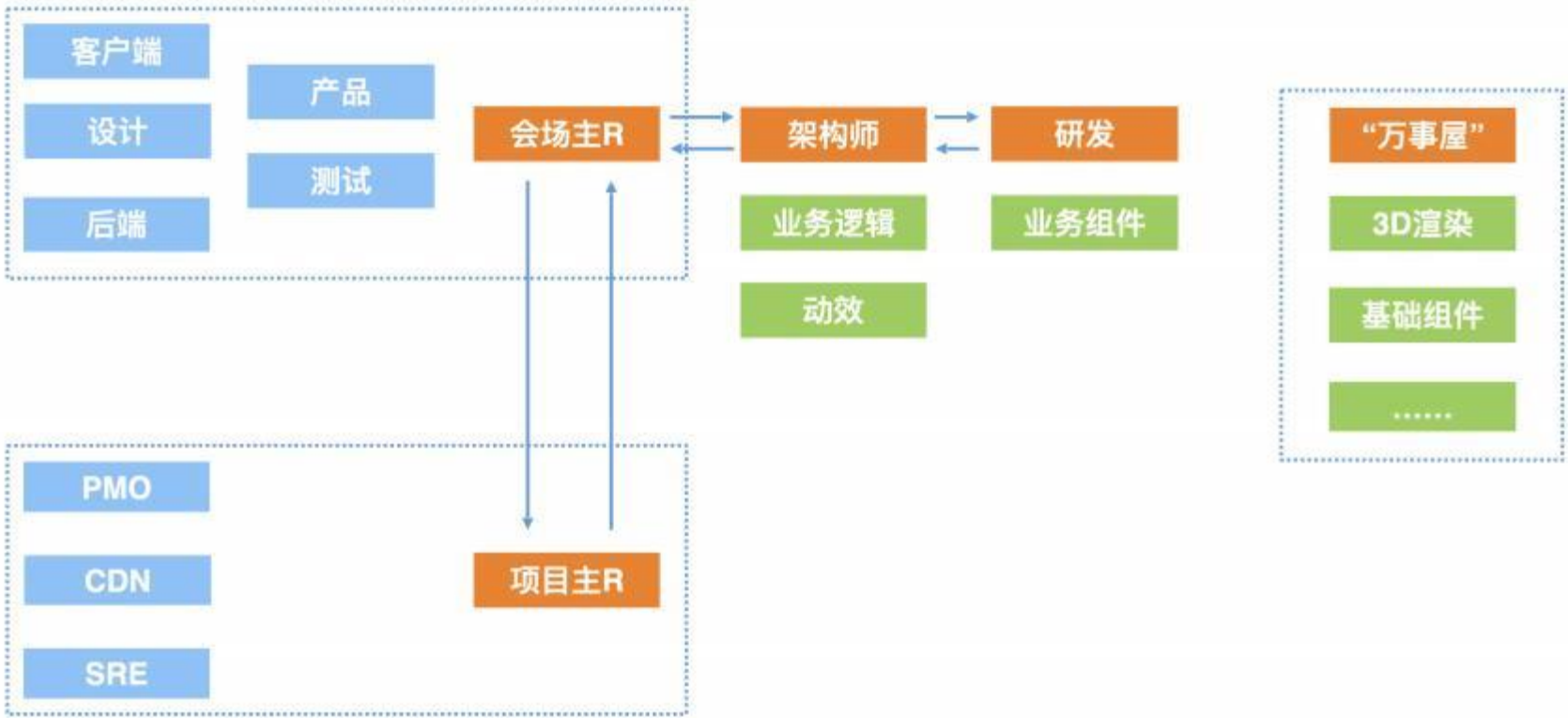

VueJS
Other


有 C 端经验
只有 B 端客户端经验


有 Vue 或者 C 端或者 TypeScript 经验
都无

问题： 无经验的占比较高， 如何保证交付？

解决方案： 职责分工


客户端
设计
后端
产品
测试
会场主R
架构师
研发
业务逻辑
业务组件
动效
“万事屋”
3D渲染
基础组件
……
PMO
CDN
SRE
项目主R

举一个例子

会场架构师：

- 页面容器组件
- 服务端接口
- 全局 model
- 拆分细粒度组件音

业务开发者：

- 实现组件
- 修改组件输入输出
- 对接公共组件

问题：

如何保证代码质量？

小老虎抽奖机：

- expose
  - s ha ke:
    `()=>void`

光带：

- props
  - text: `string` 选号机：

- props
  - i nitia lVal :
- `Nums` emit
  - c hoose:
- `Nums`
  - expose
    - scrollTo: `(val: Nums) => void`

代码质量

人工方法

- Coqe be^!eM
- 销V 工程师回测

机械方法

- nu!车」e2车
- E：E」e2车
- 」入be Cpec
- 「

!u车

目录

Volar

乡 Explore high- performance tooling for Vue

- Vue Language Features
- TypeScri pt Vue P l ugi n
- vue- tsc
- vue- component- meta
- vite- p l ugi n-vue-component-preview

Volar

乡 Explore high- performance tooling for Vue

- props
- emits
- s lots
- expose
- i
- njection
- directiv
- e

Props 的声明方法 – `defineComponent` `PropType`

```
import { defineComponent, type PropType } from 'vue';

const HelloWorld = defineComponent({
    props: {
        msg: {
            type: String as PropType< 'foo' | 'bar'>,
            required: false,
        },
    },
});
```

Props 的声明方法 – `<script setup>` `defineProps`

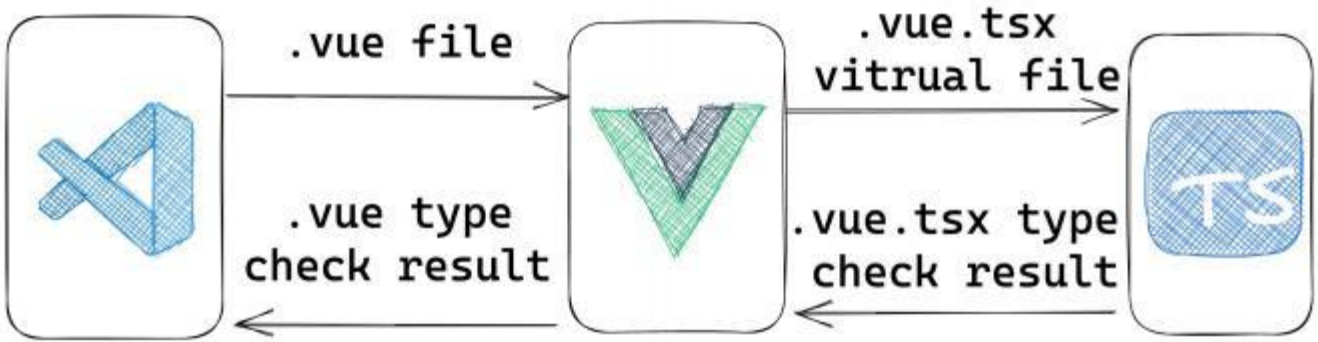
```
<script setup lang="ts">
export type HelloWorldProps = {
  msg?: 'foo' | 'bar';
}
const props = defineProps<HelloWorldProps>();
</script>
```

Props 的类型识

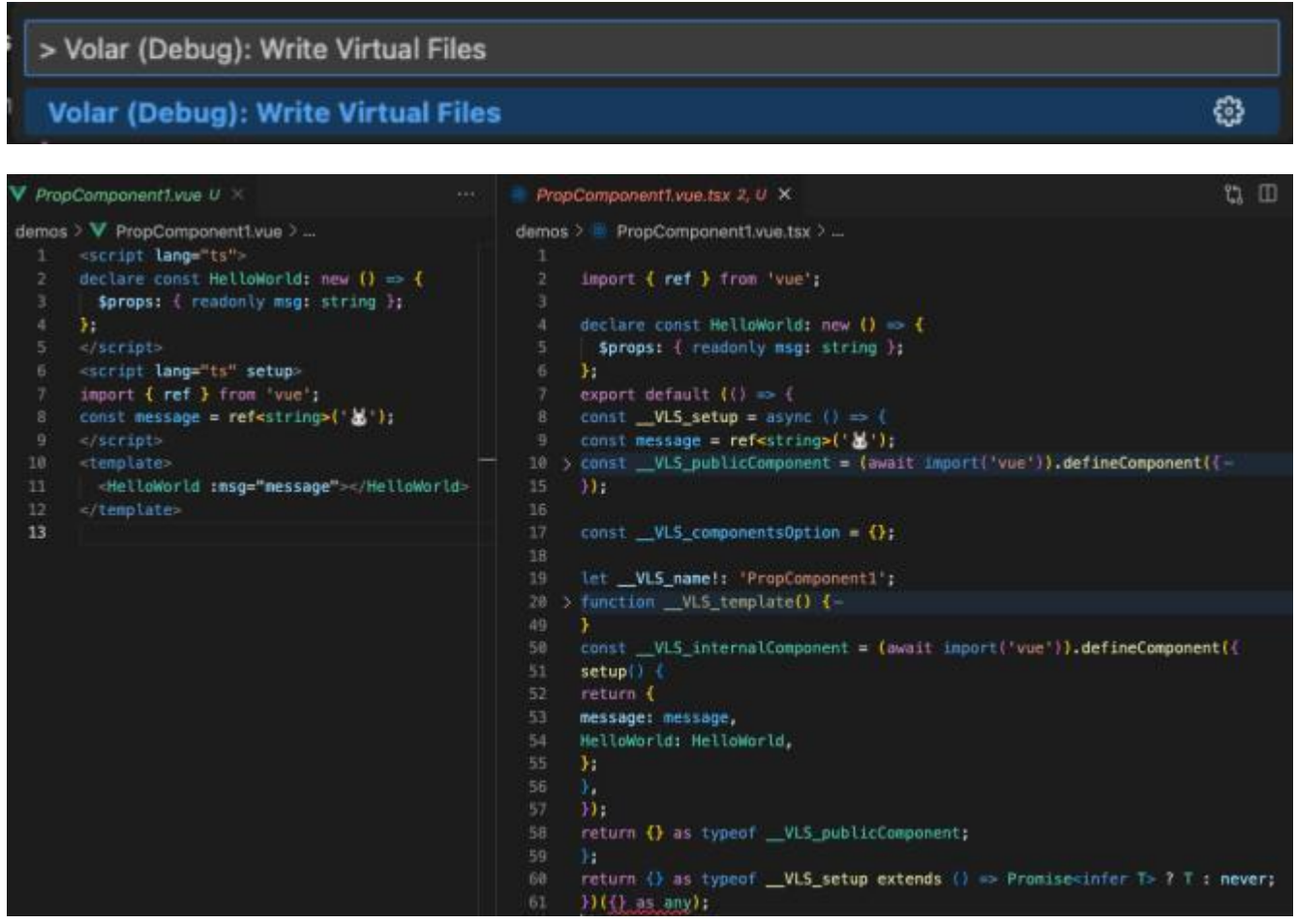
```vue
<script lang="ts">
declare const HelloWorld: new () => {
    $props: { readonly msg: string };
};
</script>
<script lang="ts" setup>
import { ref } from 'vue';
const message = ref<string>('🐰');
</script>
<template>
    <HelloWorld :msg="message"></HelloWorld>
</template>
```

```
(property) msg: string
```

Volar 的类型检查原理



Virtual File



Virtual File – script and im port

```
import { ref } from 'vue';

declare const HelloWorld: new () => {
    $props: { readonly msg: string };
};
export default ((() => {
    const __VLS_setup = async () => {
        const message = ref<string>( ' ');
        const __VLS_publicComponent = (await import( 'vue')).defineComponent({ /* ..... */
        });

        const __VLS_componentsOption = {};

        let __VLS_name !: 'PropComponent1';
        function __VLS_template() { /* ..... */
        }
        const __VLS_internalComponent = (await import( 'vue')).defineComponent({ /* ..... */
        });
        return {} as typeof __VLS_publicComponent;
    };
    return {} as typeof __VLS_setup extends () => Promise<infer T> ? T : never;
})({} as any);
```

Source File – script and import

```
<script lang="ts">
declare const HelloWorld: new () => {
    $props: { readonly msg: string };
};
</script>
<script lang="ts" setup>
import { ref } from 'vue';
const message = ref<string>( ' ');
</script>
<template>
    <HelloWorld :msg="message" ></HelloWorld>
</template>
```

Virtual File – script and im port

```
import { ref } from 'vue';

declare const HelloWorld: new () => {
    $props: { readonly msg: string };
};
export default (() => {
    const __VLS_setup = async () => {
        const message = ref<string>( '  ');
        const __VLS_publicComponent = (await import( 'vue')).defineComponent({ /* ..... */
        });

        const __VLS_componentsOption = {};

        let __VLS_name !: 'PropComponent1';
        function __VLS_template() { /* ..... */
        }
        const __VLS_internalComponent = (await import( 'vue')).defineComponent({ /* ..... */
        });
        return {} as typeof __VLS_publicComponent;
    };
    return {} as typeof __VLS_setup extends () => Promise<infer T> ? T : never;
})({} as any);
```

Source File - script setup body

```
<script lang="ts">
declare const HelloWorld: new () => {
    $props: { readonly msg: string };
};
</script>
<script lang="ts" setup>
import { ref } from 'vue';
const message = ref<string>( ' ');
</script>
<template>
    <HelloWorld :msg="message" ></HelloWorld>
</template>
```

Virtual File - script setup body

```
import { ref } from 'vue';

declare const HelloWorld: new () => {
    $props: { readonly msg: string };
};
export default ((() => {
    const __VLS_setup = async () => {
        const message = ref<string>( ' ');
        const __VLS_publicComponent = (await import( 'vue')).defineComponent({ /* ..... */
        });

        const __VLS_componentsOption = {};

        let __VLS_name !: 'PropComponent1';
        function __VLS_template() { /* ..... */
        }
        const __VLS_internalComponent = (await import( 'vue')).defineComponent({ /* ..... */
        });
        return {} as typeof __VLS_publicComponent;
    };
    return {} as typeof __VLS_setup extends () => Promise<infer T> ? T : never;
})({} as any);
```

Source File - template

```
<script lang="ts">
declare const HelloWorld: new () => {
    $props: { readonly msg: string };
};
</script>
<script lang="ts" setup>
import { ref } from 'vue';
const message = ref<string>( '  ');
</script>
<template>
    <HelloWorld :msg="message" ></HelloWorld>
</template>
```

Virtual File - template

```ts
import { ref } from 'vue';

declare const HelloWorld: new () => {
    $props: { readonly msg: string };
};
export default ((() => {
    const __VLS_setup = async () => {
        const message = ref<string>( '  ');
        const __VLS_publicComponent = (await import( 'vue')).defineComponent({ /* ..... */
        });

        const __VLS_componentsOption = {};

        let __VLS_name !: 'PropComponent1';
        function __VLS_template() { /* ..... */
        }
        const __VLS_internalComponent = (await import( 'vue')).defineComponent({ /* ..... */
        });
        return {} as typeof __VLS_publicComponent;
    };
    return {} as typeof __VLS_setup extends () => Promise<infer T> ? T : never;
})({} as any);
```

Virtual File – template - __VLS_template

+二=)口口o= II>口小I口山三负-山口山一大 口

-山口 II>口小)口根 建… 口构…

常* …… *常

-山口 II>口小I口山三负-山口山李o三负o=山=口小 建… 口

工山--o二o：-可 … 口三负o：口一 - .常II>口小I口>负山小 .下小 - 大 .增山口李o三负o=山=口小V口>负山o+ II>口小I)o三负o=山=口小 - -工山--o二o：-可 -^…

构…

常* …… *常

口

VII>口小I口山三负-山口山李o三负o=山=口小 .工山--o二o：-可 三小训"口II>口小I)口根.三山小小山训山构^V常II>口小I口山三负-山口山李o三负o=山=口小 .工山--o二o：-可^…

常* …… *常

构

常* …… *常

构

目录

进入 CN人 开发中的现状

部分Vue类型检查功能	Vue 2	Vue 3
props 检查		
emit 检查		
directive 检查		
slot 的基本类型检查		
defi neExpose 导出		

emit 的类型检查

emit 类型检查失效例

components > HelloWorld.vue > ...

```vue
<script lang="ts" setup>
export type HelloWorldEmits = {
  (event: 'msg-click', payload: MouseEvent): void;
};
const emit = defineEmits<HelloWorldEmits>();
</script>
```

emit 类型检查失效例子 （续)

```vue
<script setup lang="ts">
import HelloWorld from './components/HelloWorld.vue';
const console = globalThis.console
</script>
<template>
  <HelloWorld
    @msg-click="console.log($event)"
  />
</template>
```

```
(parameter) $event: any
```

构造类型识别例子

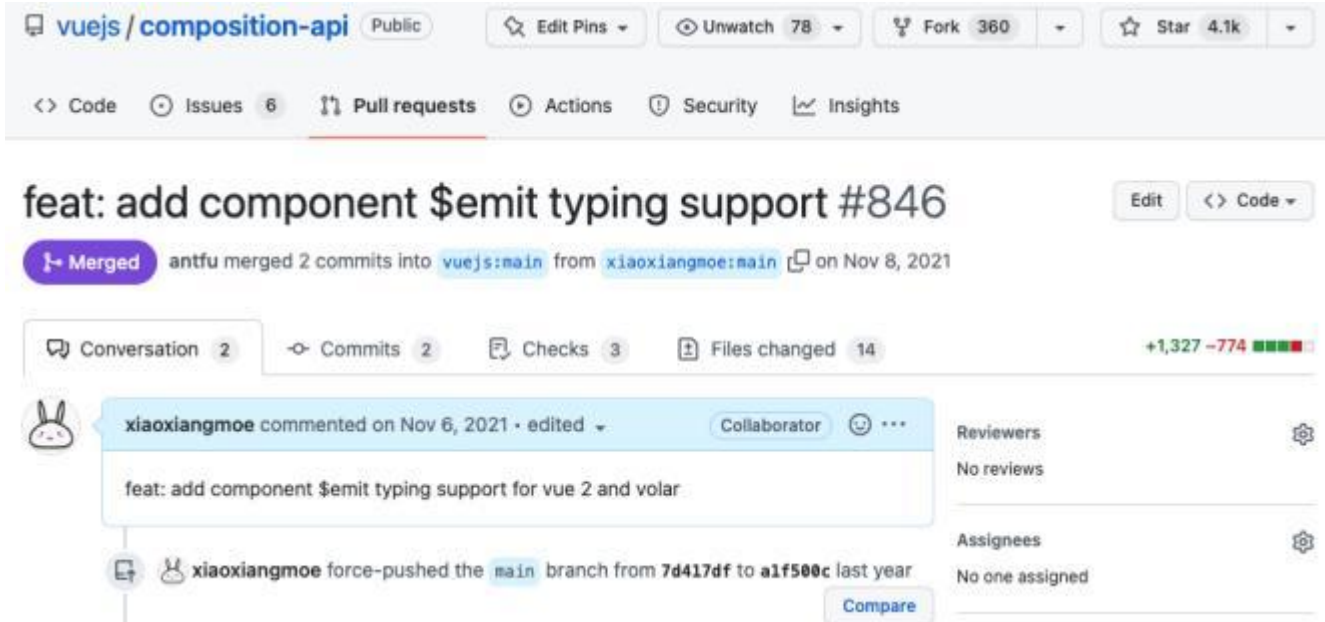
```
<script lang="ts">
declare const HelloWorld: new () => {
    $props: {};
    $emit: {
        (event: 'msg-click', payload: MouseEvent): void;
    };
};
</script>
<script setup lang="ts">
const console = globalThis.console;
</script>
<template>
    <HelloWorld @msg-click="console.log($event)" />
</template>
```

emit 的错误定位

```
import { defineComponent } from '@vue/composition-api';

const HelloWorld = defineComponent({
    props: {},
    emits: {
        'msg-click'(payload: MouseEvent) {
            return true;
        },
    },
});
type HelloWorld = InstanceType<typeof HelloWorld>;
type HelloWorldEmits = InstanceType<typeof HelloWorld>[ '$emit'];
//   ^? type HelloWorldEmits = (event: string, ...args: any[]) => Vue
```

修改


vuejs / composition-api
Public
Edit Pins
Unwatch 78
Fork 360
Star 4.1k
Code
Issues 6
Pull requests
Actions
Security
Insights
feat: add component $emit typing support #846
Edit
Code
Merged
antfu merged 2 commits into vuejs:main from xiaoxiangmoe:main on Nov 8, 2021
Conversation 2
Commits 2
Checks 3
Files changed 14
+1,327 −774
xiaoxiangmoe commented on Nov 6, 2021 • edited
Collaborator
feat: add component $emit typing support for vue 2 and volar
xiaoxiangmoe force-pushed the main branch from 7d417df to a1f500c last year
Compare
Reviewers
No reviews
Assignees
No one assigned

emit 的问题修复

```ts
import { defineComponent } from '@vue/composition-api';

const HelloWorld = defineComponent({
  props: {},
  emits: {
    'msg-click'(payload: MouseEvent) {
      return true;
    },
  },
});
type HelloWorld = InstanceType<typeof HelloWorld>;
type HelloWorldEmits = InstanceType<typeof HelloWorld>['$emit'];
//   ^? type HelloWorldEmits = (event: "msg-click", payload: MouseEvent) => Vue
```

emit 类型检查成功的例

components > HelloWorld.vue > ...

```vue
<script lang="ts" setup>
export type HelloWorldEmits = {
  (event: 'msg-click', payload: MouseEvent): void;
};
const emit = defineEmits<HelloWorldEmits>();
</script>
```

emit 类型检查成功的例子 （续)

```vue
<script setup lang="ts">
import HelloWorld from './components/HelloWorld.vue';
const console = globalThis.console
</script>
<template>
  <HelloWorld
    @msg-click="console.log($event)"
  />
</template>
```

```
(parameter) $event: MouseEvent
```

完成一系列开发后

部分Vue类型检查功能	Vue 2	Vue 3
props 检查		
emit 检查		
directive 检查		
s lot 的基本类型检查		
defi neExpose 导出		

a tool for software development using the contin uous methodo logies.

对比

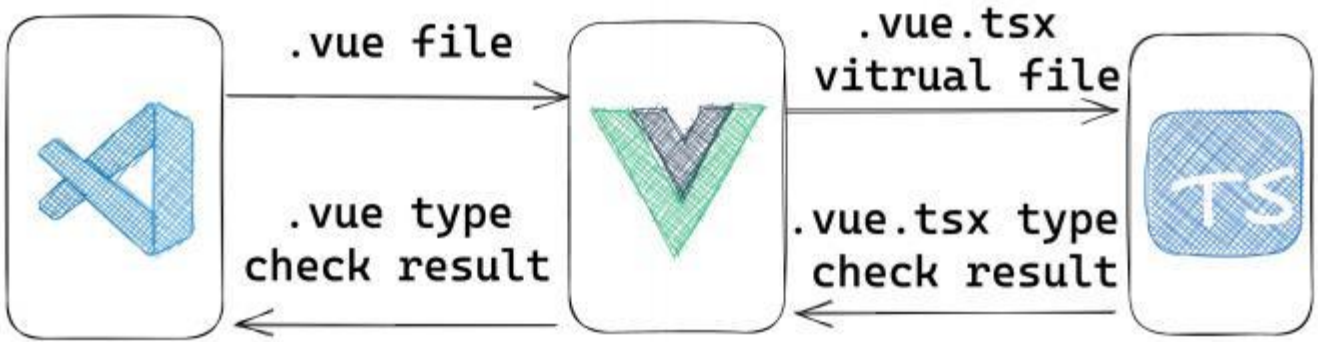
2021年:

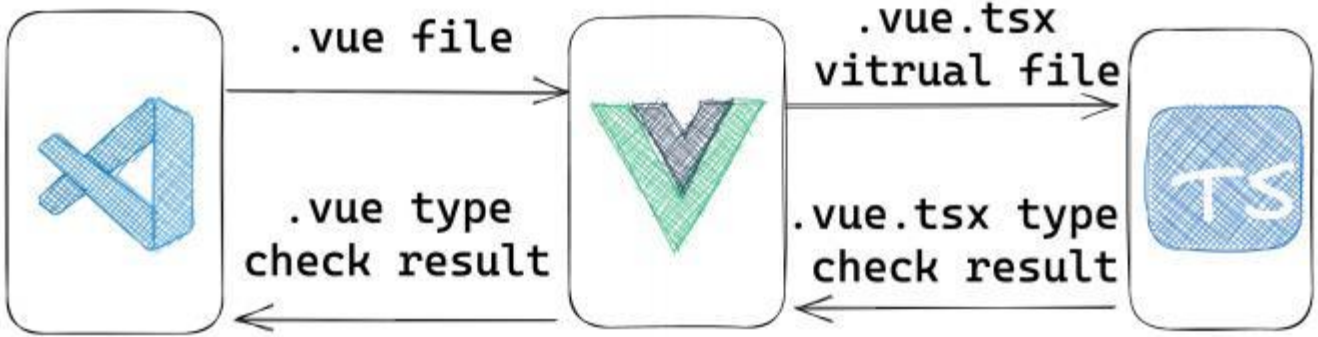
2022年:

- 2021年预热线主会场： 500 个 b ug
- 2022年预热线主会场： 700 个 b ug （三倍的代码和 素材)

目录

回顾 virtual file





Vue MACROS

MDX

GMTC
全球大前端技术大会

THANKS