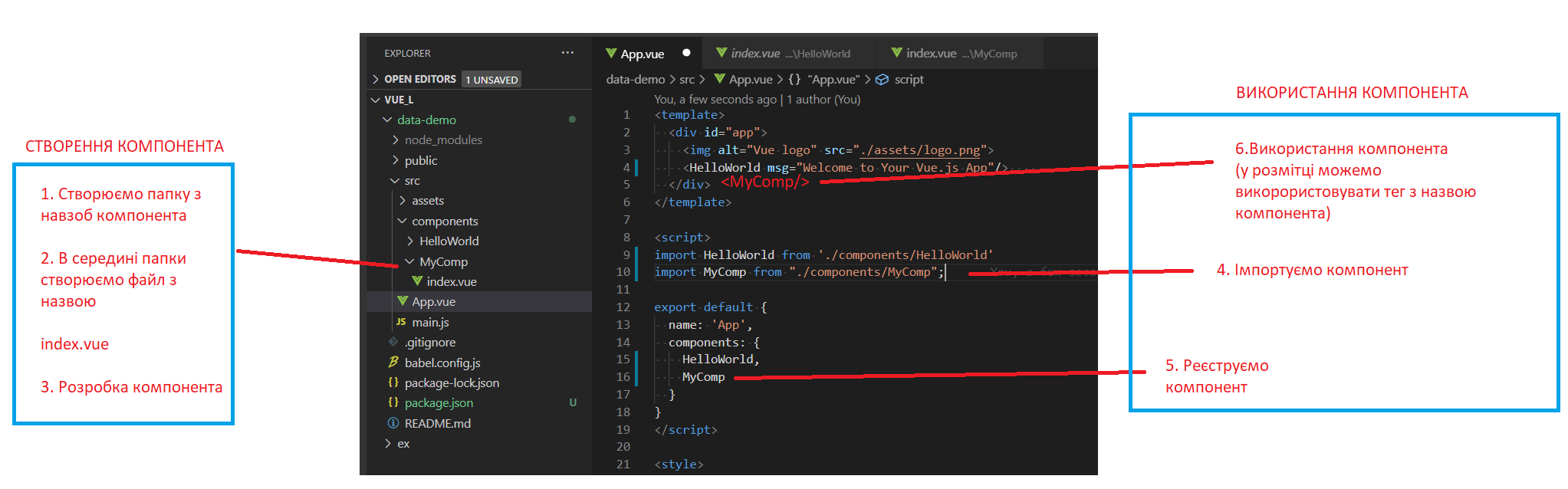
1) Схема створення компонента (файлу компонента) та підключення компонента



2) Розробка презентаційного компонента

Дані, які необхідні для роботи компонента, вказуємо у розділі “props” (можемо використати сніпет – “vprops”)

|  |  |
| --- | --- |
| Загальний вигляд | Приклад |
| <!-- vbase -->  <template>  <div>  . . .  </div>  </template>  <script>  export default {  props: {  назва\_властивості: {  type: тип\_властивості,  default: значення\_за\_замовчуванням  },  назва\_властивості2: {  type: тип\_властивості2,  default: значення\_за\_замовчуванням2  },  . . .  }, | //Що нам потрібно мати для роботи компонента    props: {      imgSrc: {        type: String,        default:          'https://thumbs.dreamstime.com/b/no-image-available-icon-flat-vector-no-image-available-icon-flat-vector-illustration-132482930.jpg',      },      caption: {        type: String,        default: 'No caption',      },      clicksNumber: {        type: Number,        default: 0,      },    }, |

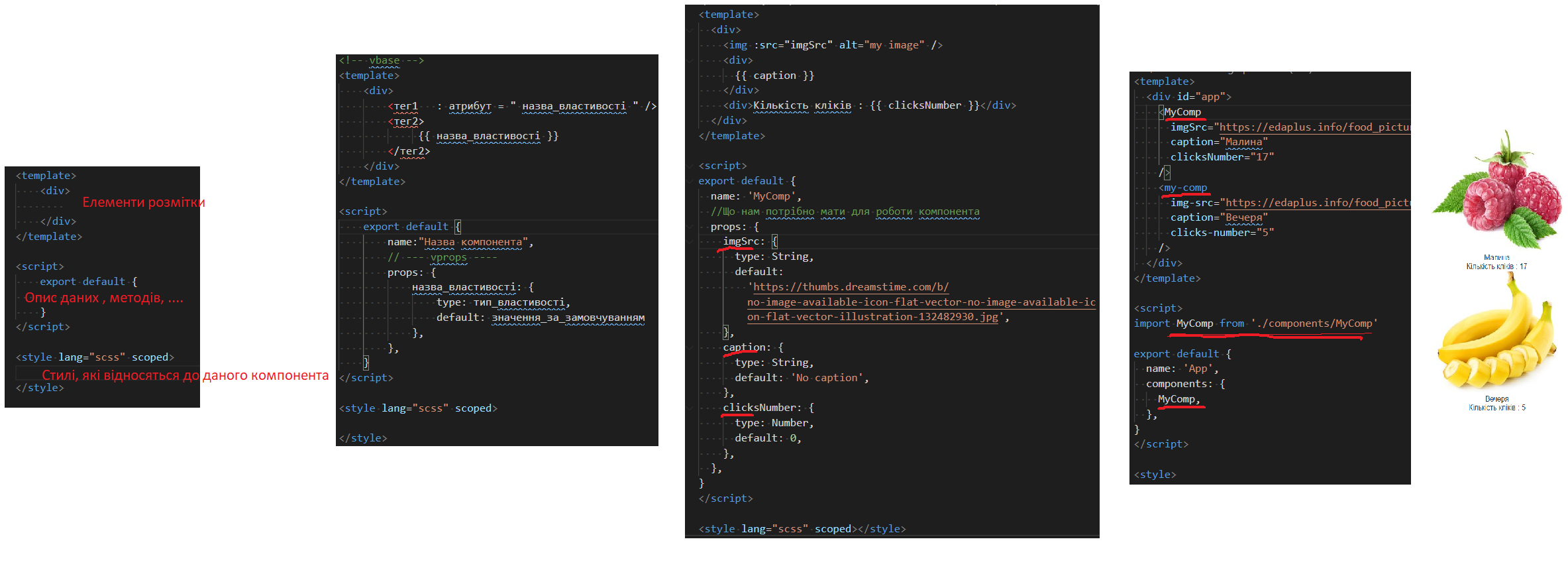
Для того, щоб використати дані у розмітці треба дотримуватися правил:

1)значення атрибуту тегу

|  |  |
| --- | --- |
| Загальний вигляд | Приклад |
| <template>  <div>  . . .  <тег : атрибут = " назва\_властивості " />  . . .  </div>  </template>  <script>  export default {  props: {  назва\_властивості: {  type: тип\_властивості,  default: значення\_за\_замовчуванням  },  . . .  },  . . .  </script> | <template>    <div>      <img :src="imgSrc" alt="my image" />    </div>  </template>  <script>  export default {    name: 'MyComp',    //Що нам потрібно мати для роботи компонента    props: {      imgSrc: {        type: String,        default:          'https://thumbs.dreamstime.com/b/no-image-available-icon-flat-vector-no-image-available-icon-flat-vector-illustration-132482930.jpg',      },  ...  } |

2)внутрішній контент тегу

|  |  |
| --- | --- |
| Загальний вигляд | Приклад |
| <template>  <div>  . . .  <тег>  {{ назва\_властивості }}  </тег>  . . .  </div>  </template>  <script>  export default {  props: {  назва\_властивості: {  type: тип\_властивості,  default: значення\_за\_замовчуванням  },  . . .  },  . . .  </script> | <template>    <div>      ...      <div>        {{ caption }}      </div>      <div>  Кількість кліків : {{ clicksNumber }}  </div>    </div>  </template>  <script>  export default {    name: 'MyComp',    //Що нам потрібно мати для роботи компонента    props: {  ...      caption: {        type: String,        default: 'No caption',      },      clicksNumber: {        type: Number,        default: 0,      },    },  }  </script> |



3) Використання внутрішніх даних компонента. Внутрішні дані компонента можна зберігати у “data”. Використання у розмітці аналогічне до властивостей (props) за назвою. Особливістю таких даних є те, що підтримується «реактивність» (якщо змінюємо значення даних – автоматично ініціюється оновлення елементів розмітки, що його містить, і ми побачимо нові дані). Для звертання до поля у методах необхідно використовувати шаблон «this. назва\_поля»

|  |  |
| --- | --- |
| Загальний вигляд | Приклад |
| <template>  <div>  . . .  <тег : атрибут = " назва\_поля " />  . . .  <тег>  {{ назва\_поля }}  </тег>  . . .  </div>  </template>  <script>  export default {  . . . .  //Внутрішні дані компонента (сніпет - vdata)  data() {  return {  назва\_поля : початкове\_значення,  назва\_поля2 : початкове\_значення2,  . . .  }  },  . . .  },  . . .  </script> | <template>    <div>      <div>        {{ randomItem }}      </div>      <button @click="getRandomItem">        {{ triggerTitle }}      </button>    </div>  </template>  <script>  export default {    name: 'RandomSelector',    //Що потрібно мати для використання (vprops)    props: {      items: {        type: Array,        default: () => [],      },      triggerTitle: {        type: String,        default: 'Go',      },    },    //Внутрішні дані компонента (vdata)    data() {      return {        randomItem: 'Do selection',      }    },      //Методи (функції) компонента (vmethod)    methods: {      getRandomItem() {        const maxIndex = this.items.length - 1        //Math.floor(min+Math.random()\*(max-min+1) )        const randIndex = Math.floor(0 + Math.random() \* (maxIndex - 0 + 1))        this.randomItem = this.items[randIndex]      },    },  }  </script>  <style lang="scss" scoped></style> |