**Литеральные типы (или псевдонимы типов type)** позволяют задать конкретные значения для переменных или параметров функций. Это обеспечивает более строгую проверку типов и предотвращает ошибки.

  type Color = "red" | "green" | "blue"; // Строковые литералы

type Color = “yellow” – ошибка, дважды объявить в одной области видимости нельзя

  const favoriteColor: Color = "red"; // Корректно

**Объединения (Union Types)**

Объединения позволяют комбинировать несколько типов в один, что делает код более гибким. Объединение обозначается символом `|`.

- **Пример использования объединений:**

let identifier: number | string; // Переменная может быть либо числом, либо строкой

identifier = 42; // Допустимо

identifier = "Hello"; // Допустимо

```

- Параметры функций с объединениями:

function display(data: number | string) {

console.log(data);

}

display(100); // Допустимо

display("Test"); // Допустимо

**Преимущества и ограничения**

- Преимущества:

- Улучшенная безопасность типов.

- Более чистый и понятный код.

- Гибкость в работе с различными типами данных.

- Ограничения:

- Сложность кода может увеличиваться при использовании множества объединений.

- Не всегда подходят для всех сценариев; иногда лучше использовать другие подходы (например, перегрузку функций).