

# Week 4

- Compound data types
- Structs, enums, unions

# Given the following task:

- You are a Fortnite developer
- You want a method that returns the current gun
- You want a method that gets a gun and does pew pew
- Each gun has ammo and damage



# Where do we have trouble?

- Returning multiple values
- Making clear some values build a bigger structure
- Structuring data in general

# Just use a class?



# OOP (Pre Alpha)

- C has no Classes
- C has no inheritance
- C has no function overloading
- C has no private/public/protected keywords

# BUT

- C has ways to combine types into a new type
- Introducing:

# STRUCT

# Combine multiple data types into a bigger one

- Concert ticket: Date, Time, Location and Band
- Date: Day, Month, Year  
Day, Month and Year: int
- Time: Hour, Minute  
Hour and minute: int
- Location: Row and Column  
Row and Column: unsigned int
- Band: Name  
Name: Some amounts of characters

# Struct

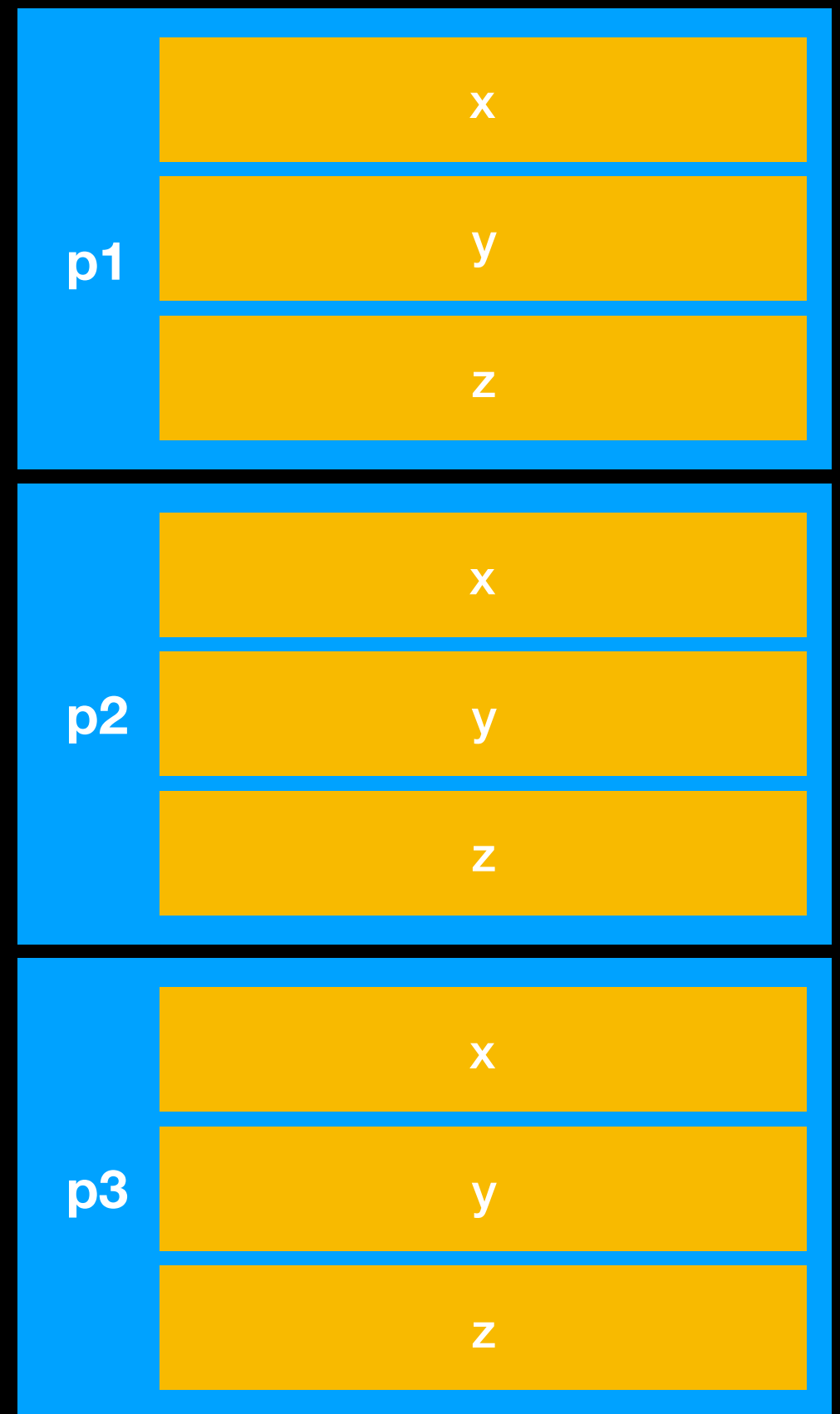
- struct is a keyword in C
- There are three ways to deal with structs, only one is used really often



# Syntax of a struct

- ```
struct [Tag] {  
    member;  
    member;  
    member;  
};
```

```
struct Point{  
    float x;  
    float y;  
    float z;  
} p1, p2, p3;
```



**Stack**

# „Named structs“

```
struct Point{  
    float x;  
    float y;  
    float z;  
};
```

```
int main(){  
    struct Point p1;  
    struct Point p2;  
    struct Point p3;
```

```
    printf("%f\n", p1.x);  
    printf("%f\n", p2.z);  
    printf("%f\n", p3.x);
```

```
    return 0;  
}
```

```
struct{  
    float x;  
    float y;  
    float z;  
} p1;
```

```
struct Point{  
    float x;  
    float y;  
    float z;  
};
```

```
struct Point p1;
```

# typedef

- Used to give other names to types
- typedef unsigned char BYTE  
BYTE b = 18;
- Can be used to create a new type from a struct

# Literally types

```
struct Point{  
    float x;  
    float y;  
    float z;  
};
```

```
int main(){  
    struct Point p1;  
    struct Point p2;  
    struct Point p3;
```

```
    printf("%f\n", p1.x);  
    printf("%f\n", p2.z);  
    printf("%f\n", p3.x);
```

```
    return 0;
```

```
}
```

```
typedef struct{  
    float x;  
    float y;  
    float z;  
} Point;
```

```
int main(){  
    Point p1;  
    Point p2;  
    Point p3;
```

```
    printf("%f\n", p1.x);  
    printf("%f\n", p2.z);  
    printf("%f\n", p3.x);
```

```
    return 0;
```

```
}
```

# Exercises

- Create a Gun struct that holds an amount of remaining ammo, ammo in the clip and damage
- Write a method to calculate how many times the gun can reload until it runs out of bullets
- Write a method that reloads the gun
- Write a „pew“ method that shoots a bullet and print the damage done

# More Exercises

- A player has a gun, a shield (0-100) and health (0-100)
- Write a method that calculates damage done to a player by a gun
- A player is able to drink a shield potion or take a medkit to increase his life, write methods for that



# Even More Exercises

- A person has some amount of money and a family is made up from 3 persons (mom, dad and a fortnite playing kid)
- The kid wants to buy skins, write a method that takes 20\$ from the fathers money away