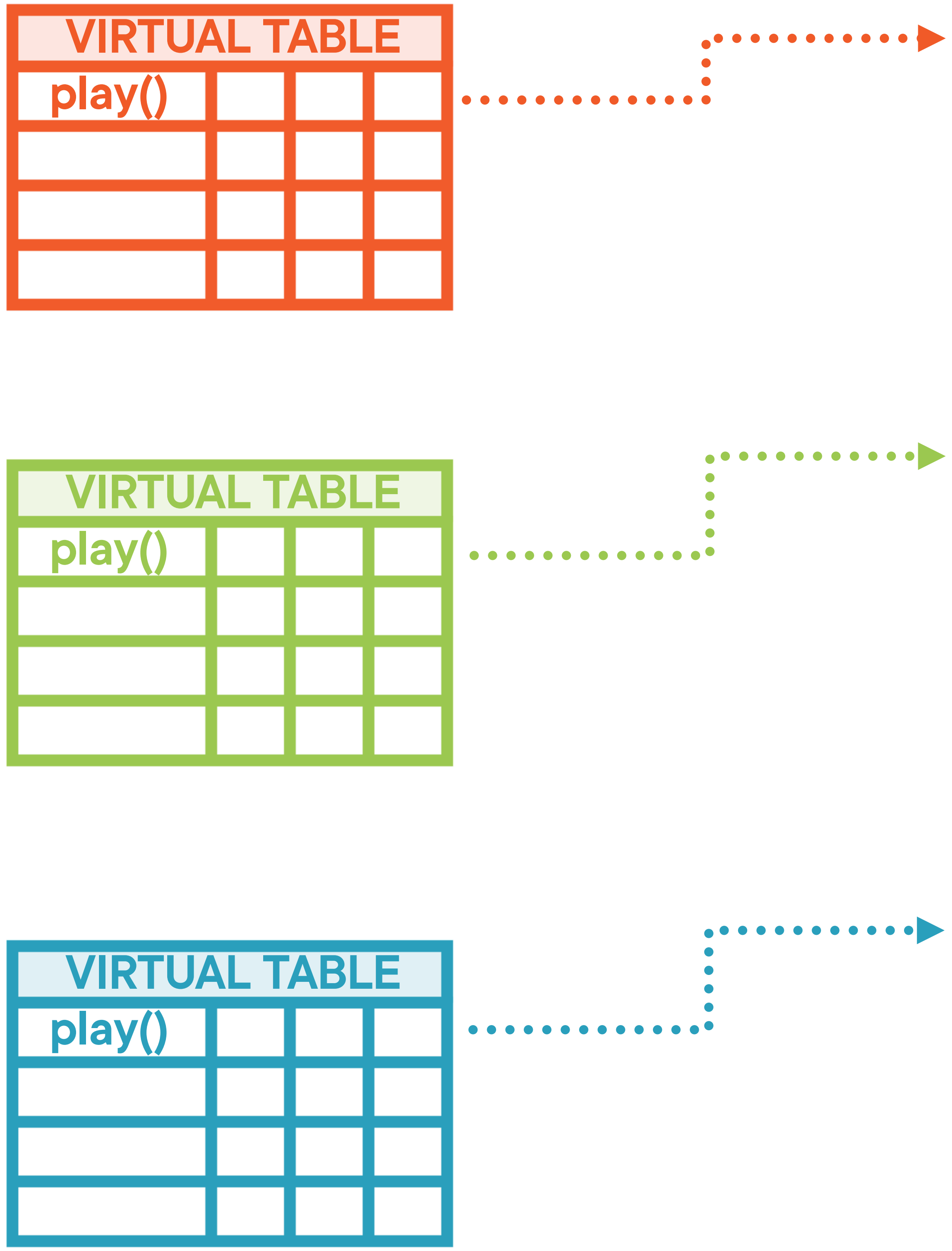
Pointing to Class Members

Mateo Prigl
Software Developer

```
class Instrument
{
  bool electric;

  (...)
};
```

```
class Guitar : public Instrument
{
  bool electric; // inherited

  int num_of_strings;
  (...)
};
```

```
class Synth : public Instrument
{
  bool electric; // inherited

  int num_of_keys;
  (...)
};
```



```
virtual void play()
{
  std::cout << "Instrument sound."
            << std::endl;
}
```

```
void play() override
{
  std::cout << "Guitar sound with "
            << num_of_strings
       << " strings." << std::endl;
}
```

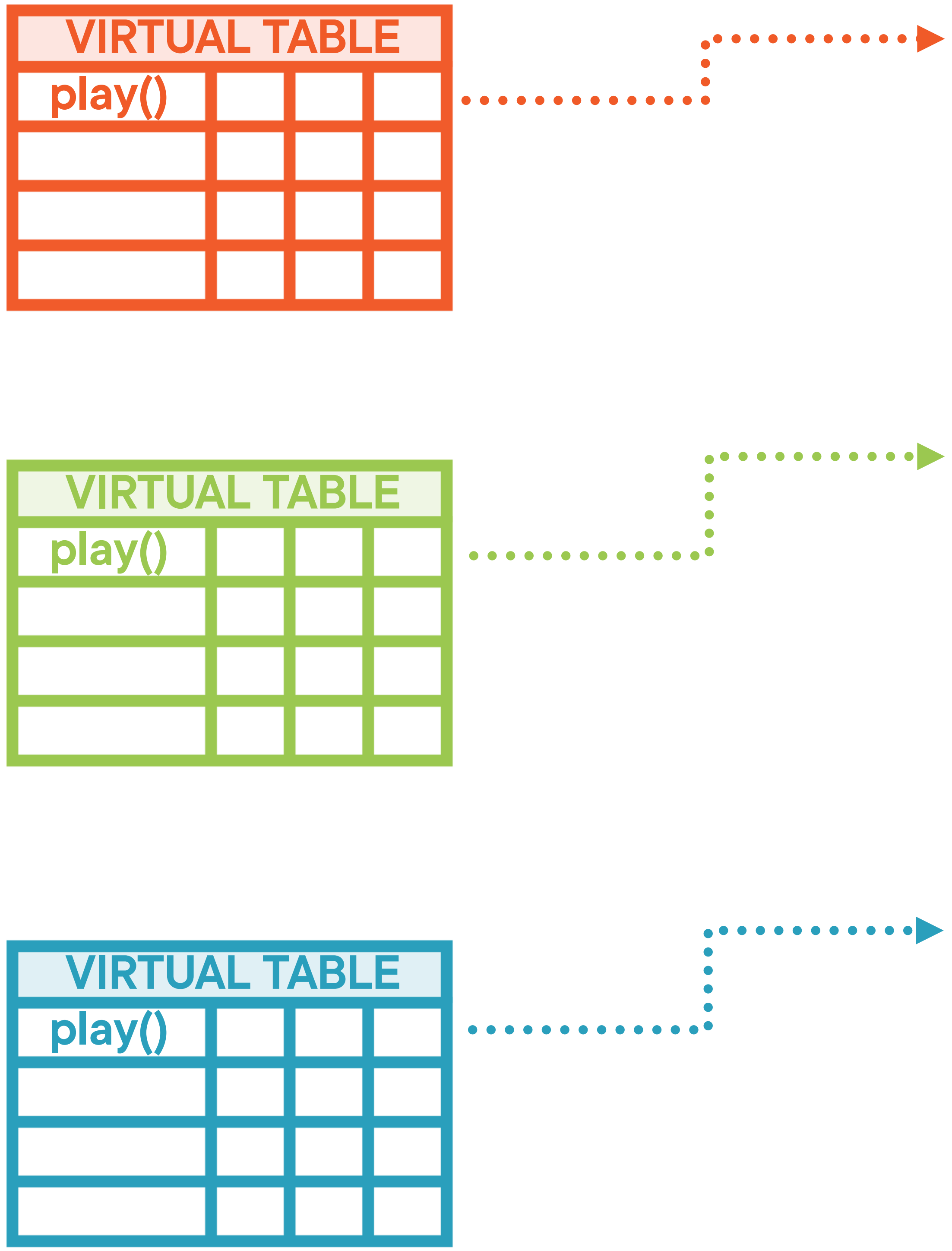
```
void play() override
{
  std::cout << "Synth sound with "
            << num_of_keys
          << " keys." << std::endl;
}
```

```
Instrument *i = new Guitar(true);
i->play();
```

```
class Instrument
{
  bool electric;

  (...)
};
```

```
class Guitar : public Instrument
{
  bool electric; // inherited

  int num_of_strings;
  (...)
};
```

```
class Synth : public Instrument
{
  bool electric; // inherited

  int num_of_keys;
  (...)
};
```



```
virtual void play()
{
  std::cout << "Instrument sound."
            << std::endl;
}
```

```
void play() override
{
  std::cout << "Guitar sound with "
            << num_of_strings
       << " strings." << std::endl;
}
```

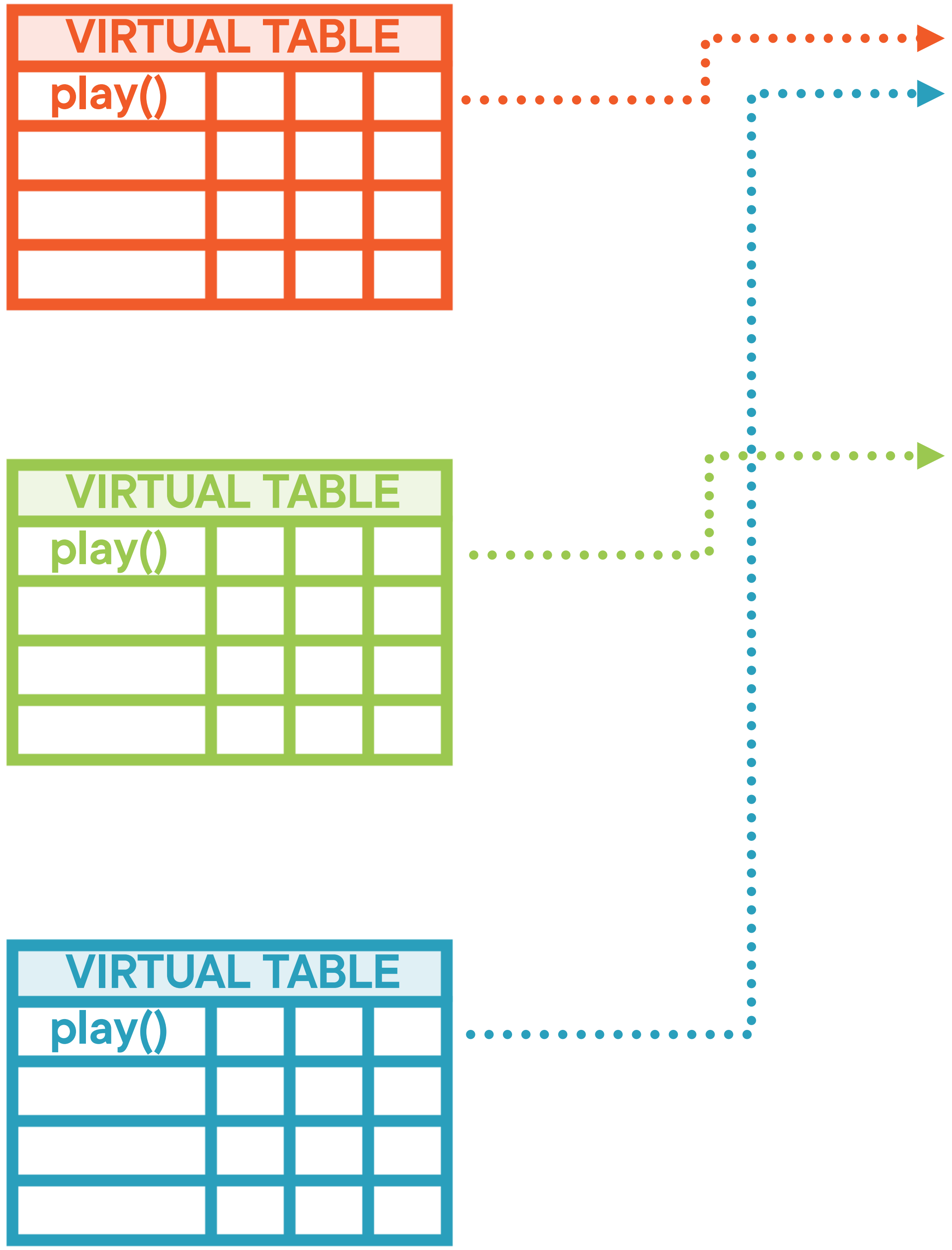
```
void play() override
{
  std::cout << "Synth sound with "
            << num_of_keys
          << " keys." << std::endl;
}
```

```
Instrument *i = new Guitar(true);
i->play();
```

```
class Instrument
{
  bool electric;

  (...)
};
```

```
class Guitar : public Instrument
{
  bool electric; // inherited

  int num_of_strings;
  (...)
};
```

```
class Synth : public Instrument
{
  bool electric; // inherited

  int num_of_keys;
  (...)
};
```



```
virtual void play()
{
  std::cout << "Instrument sound."
            << std::endl;
}
```

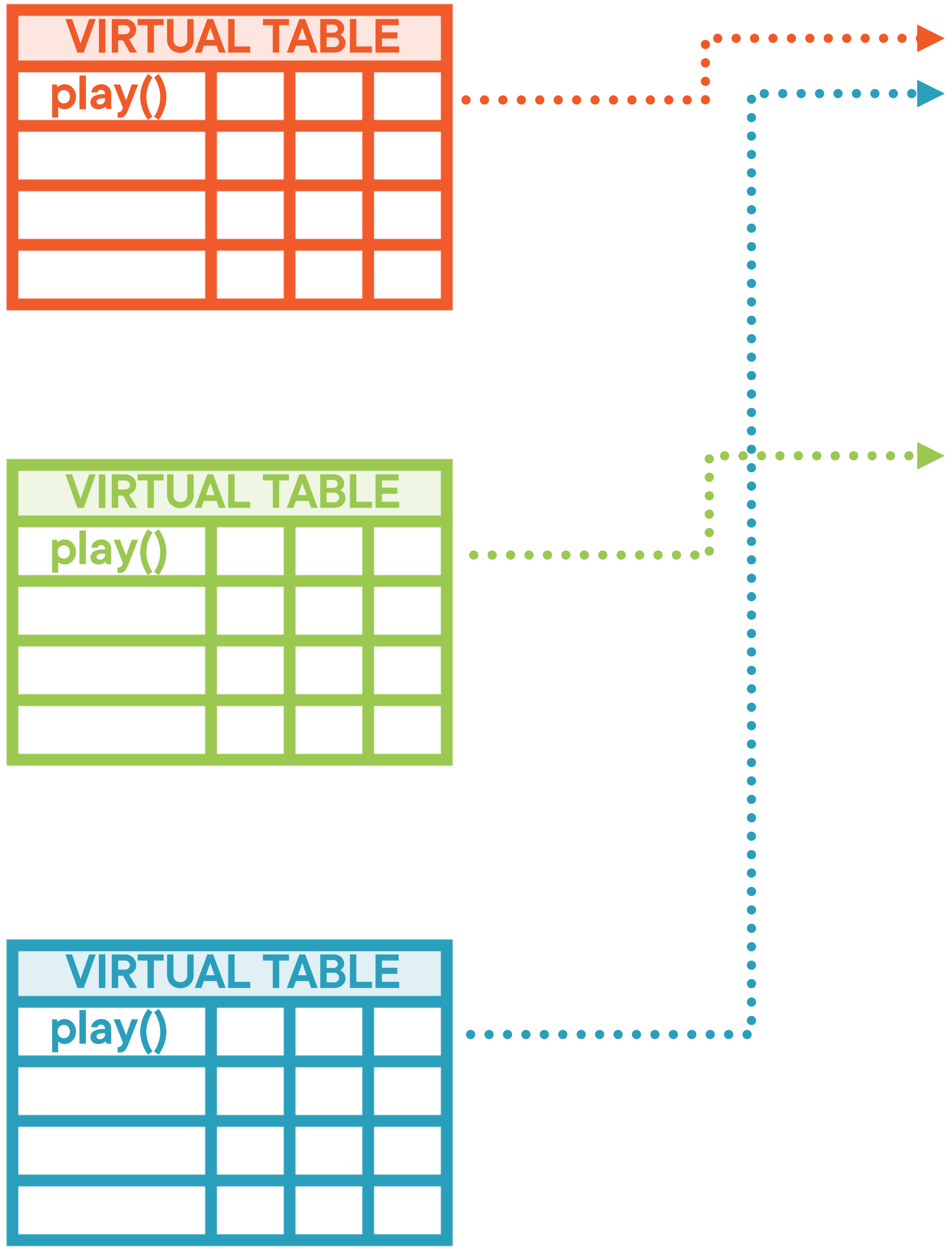
```
void play() override
{
  std::cout << "Guitar sound with "
            << num_of_strings
        << " strings." << std::endl;
}
```

```
Instrument *i = new Synth(true);
i->play();
```

```cpp
class Instrument
{
  bool electric;

  (...)
};
```

```cpp
class Guitar : public Instrument
{
  bool electric; // inherited

  int num_of_strings;
  (...)
};
```

```cpp
class Synth : public Instrument
{
  bool electric; // inherited

  int num_of_keys;
  (...)
};
```



```cpp
virtual void play()
{
  std::cout << "Instrument sound."
            << std::endl;
}
```

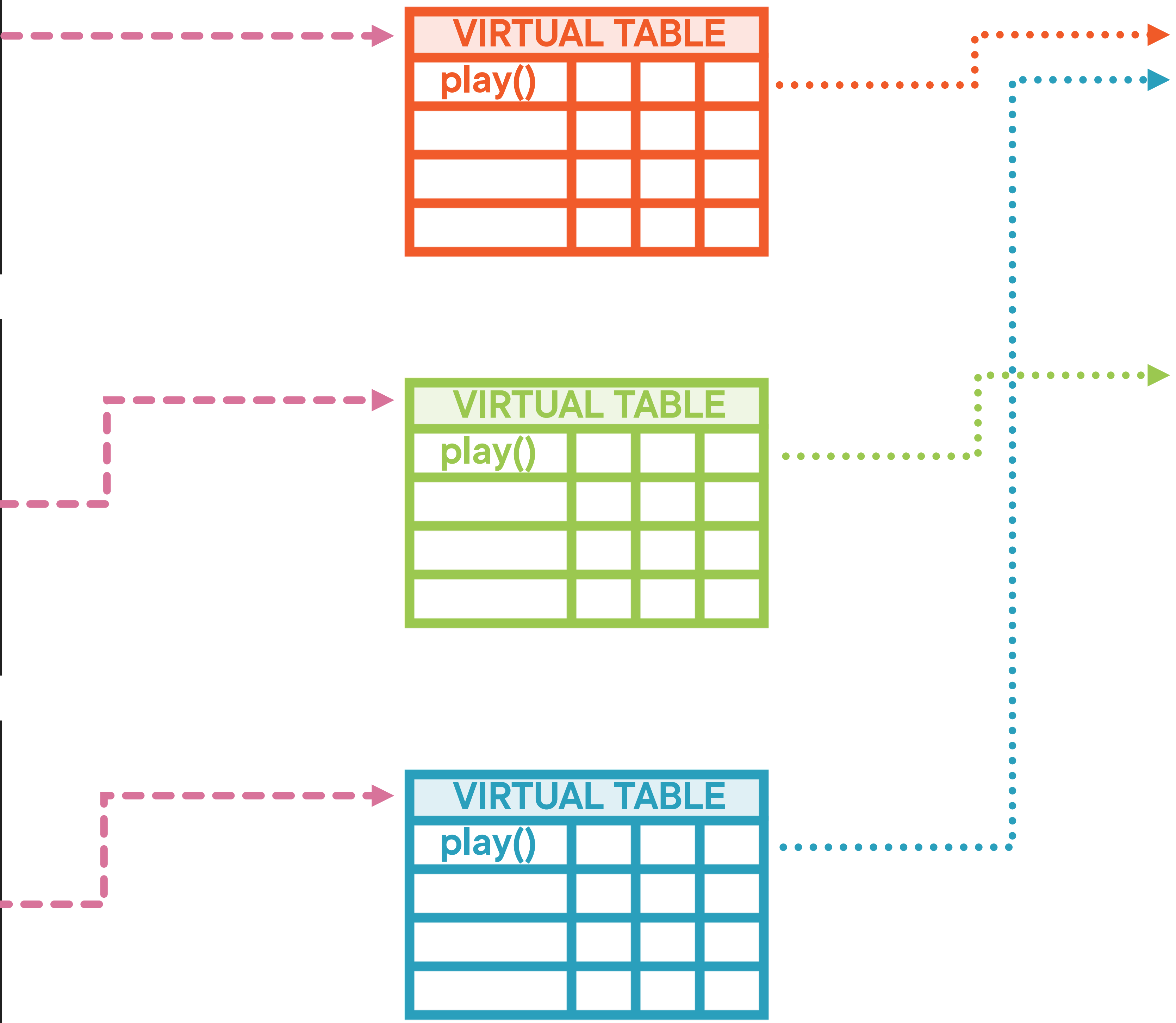
```cpp
void play() override
{
  std::cout << "Guitar sound with "
           << num_of_strings
        << " strings." << std::endl;
}
```

```cpp
Instrument *i = new Synth(true);
i->play();
```

```
class Instrument
{
  bool electric;
  *__vptr;
  (...)
};
```

```
class Guitar : public Instrument
{
  bool electric; // inherited
  *__vptr; // inherited
  int num_of_strings;
  (...)
};
```

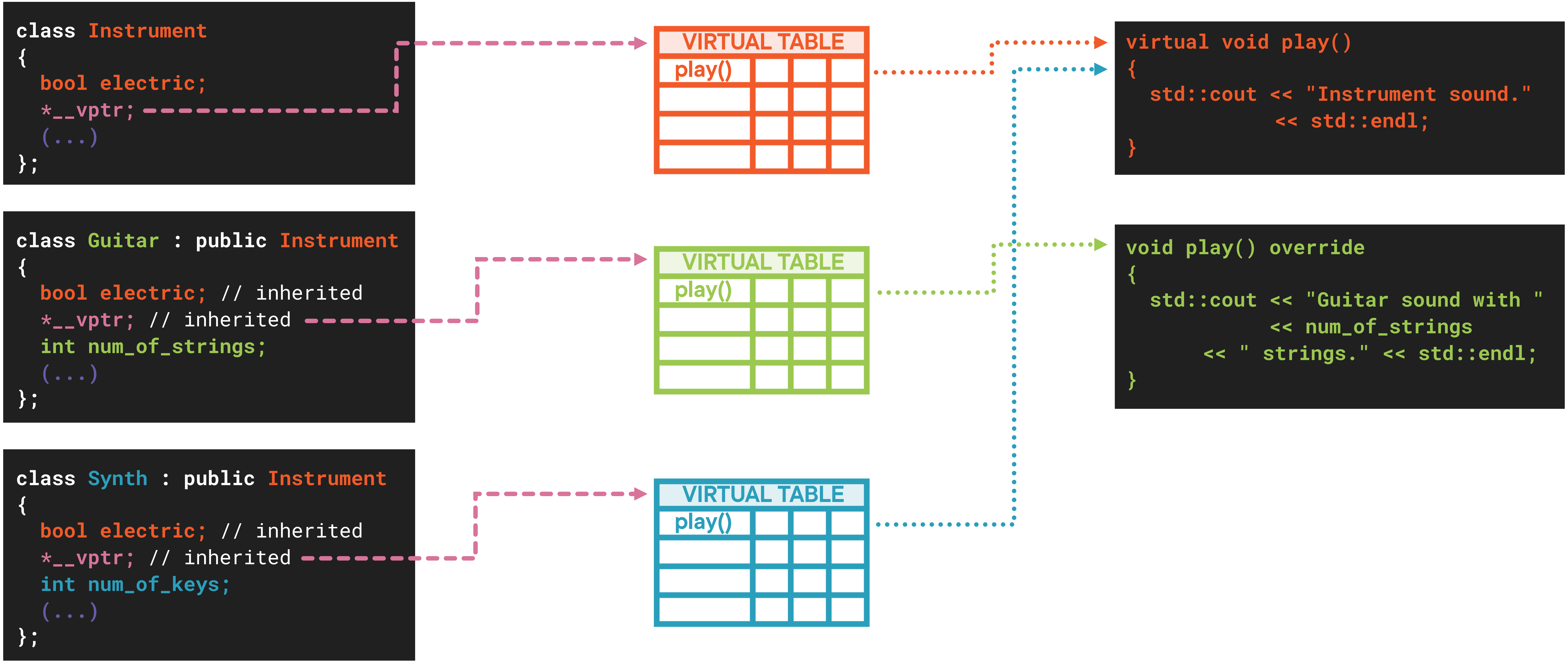
```
class Synth : public Instrument
{
  bool electric; // inherited
  *__vptr; // inherited
  int num_of_keys;
  (...)
};
```



```
virtual void play()
{
  std::cout << "Instrument sound."
            << std::endl;
}
```

```
void play() override
{
  std::cout << "Guitar sound with "
            << num_of_strings
      << " strings." << std::endl;
}
```

```
Instrument *i = new Synth(true);
i->play();
```



```
class Instrument
{
  bool electric;
  *__vptr;
  (...)
};
```

```
class Guitar : public Instrument
{
  bool electric; // inherited
  *__vptr; // inherited
  int num_of_strings;
  (...)
};
```

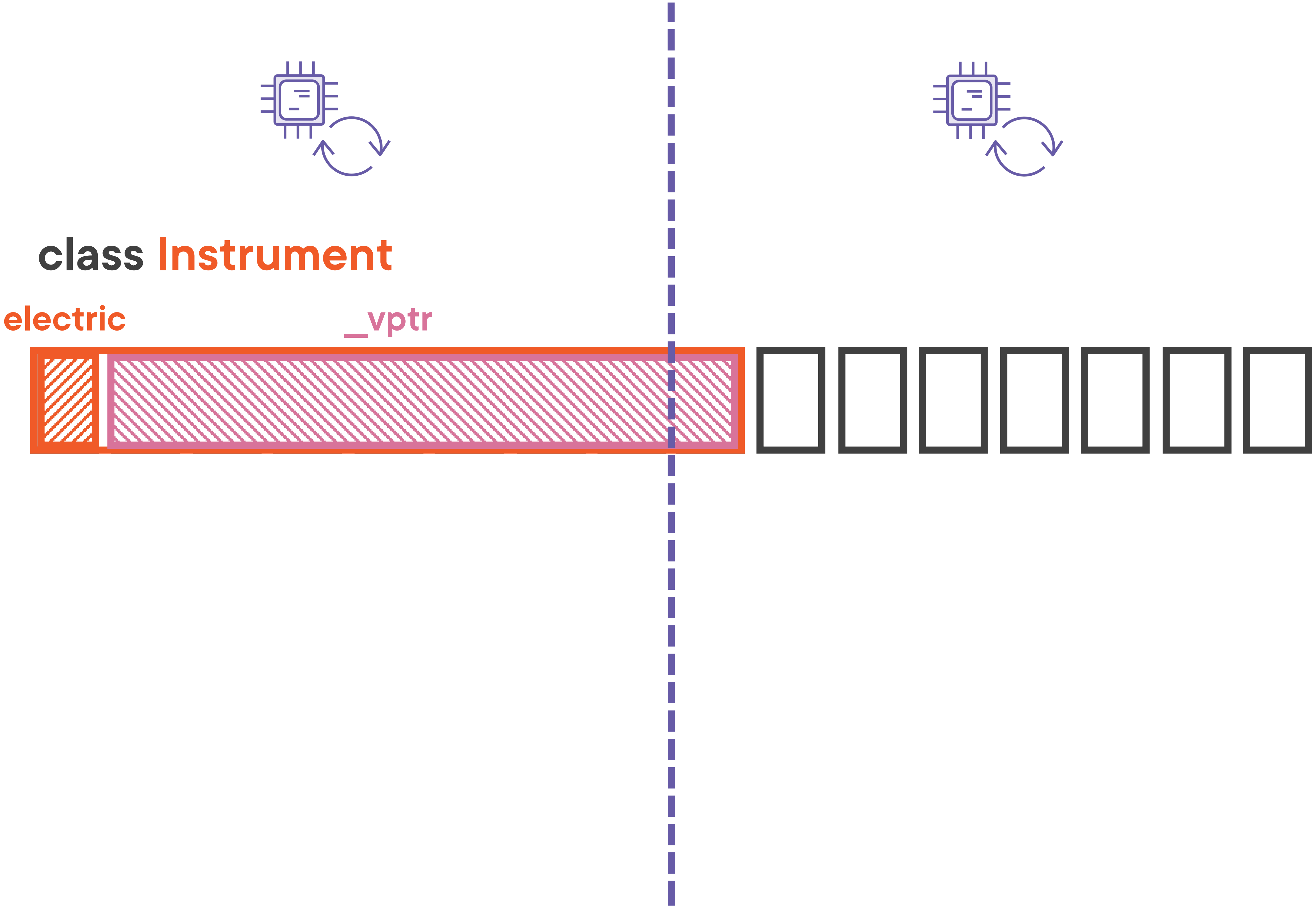
```
class Synth : public Instrument
{
  bool electric; // inherited
  *__vptr; // inherited
  int num_of_keys;
  (...)
};
```

```
virtual void play()
{
  std::cout << "Instrument sound."
            << std::endl;
}
```

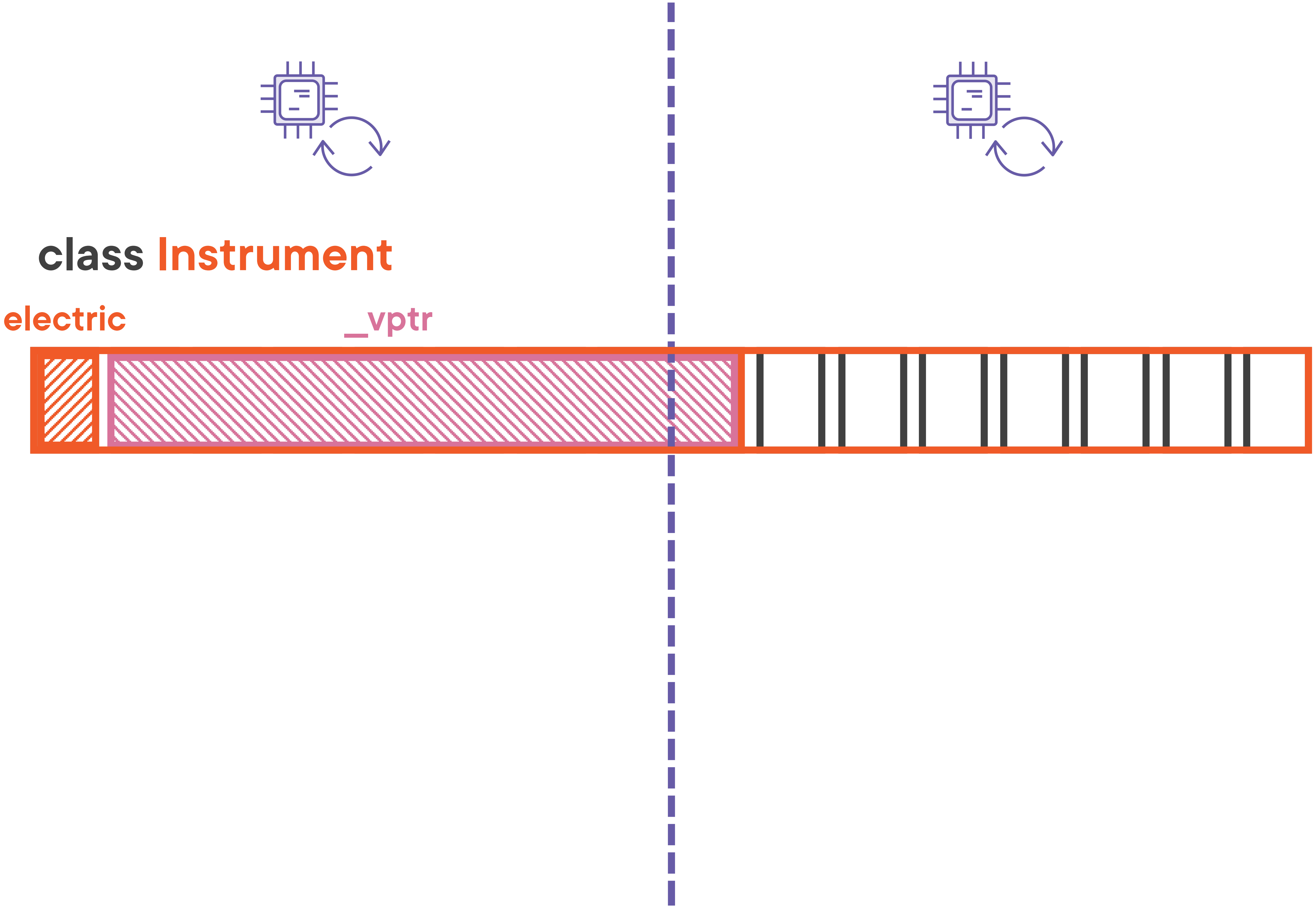
```
void play() override
{
  std::cout << "Guitar sound with "
          << num_of_strings
       << " strings." << std::endl;
}
```

```
Instrument *i = new Guitar(true);
i->play();
```

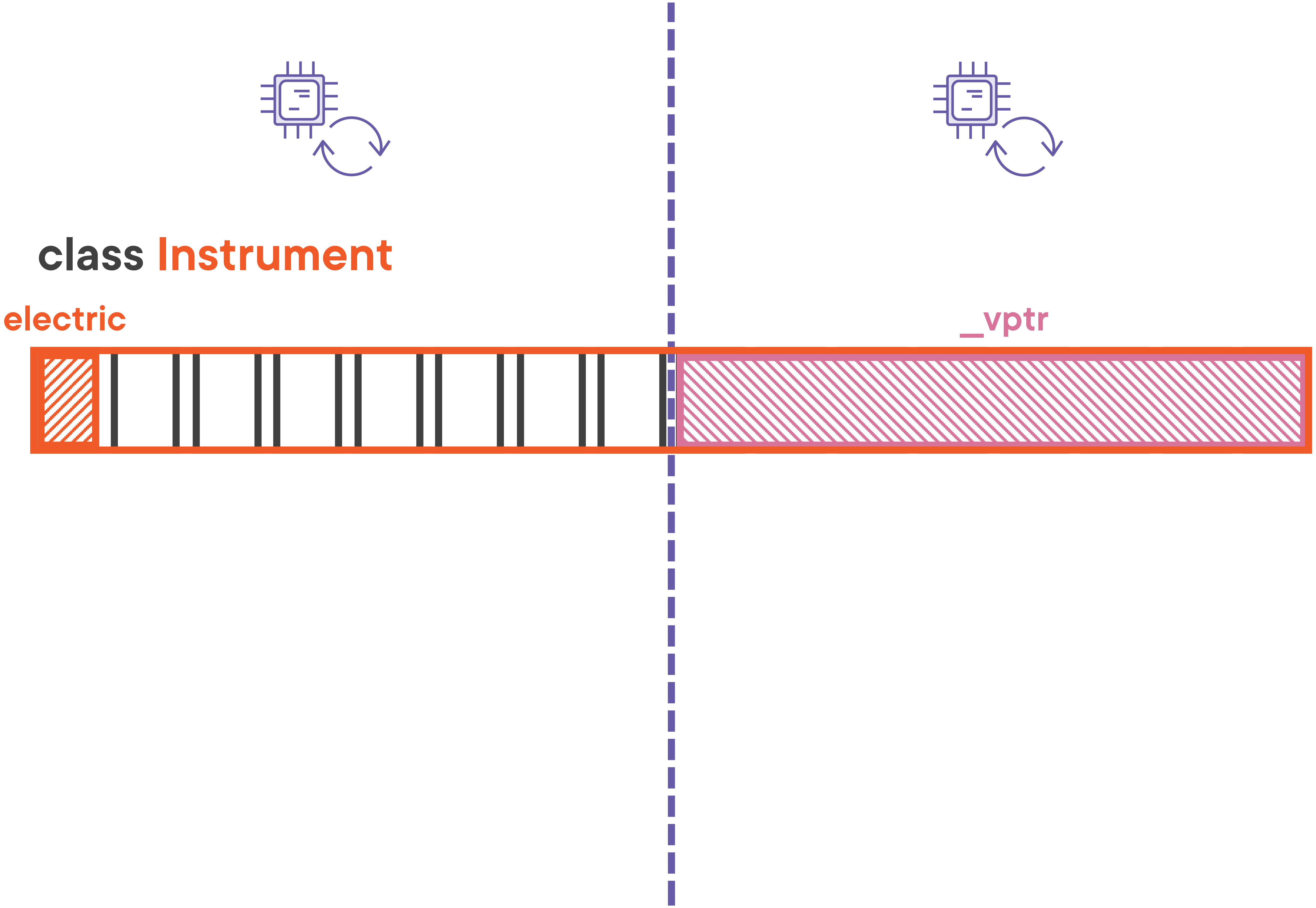
Data Structure Alignment



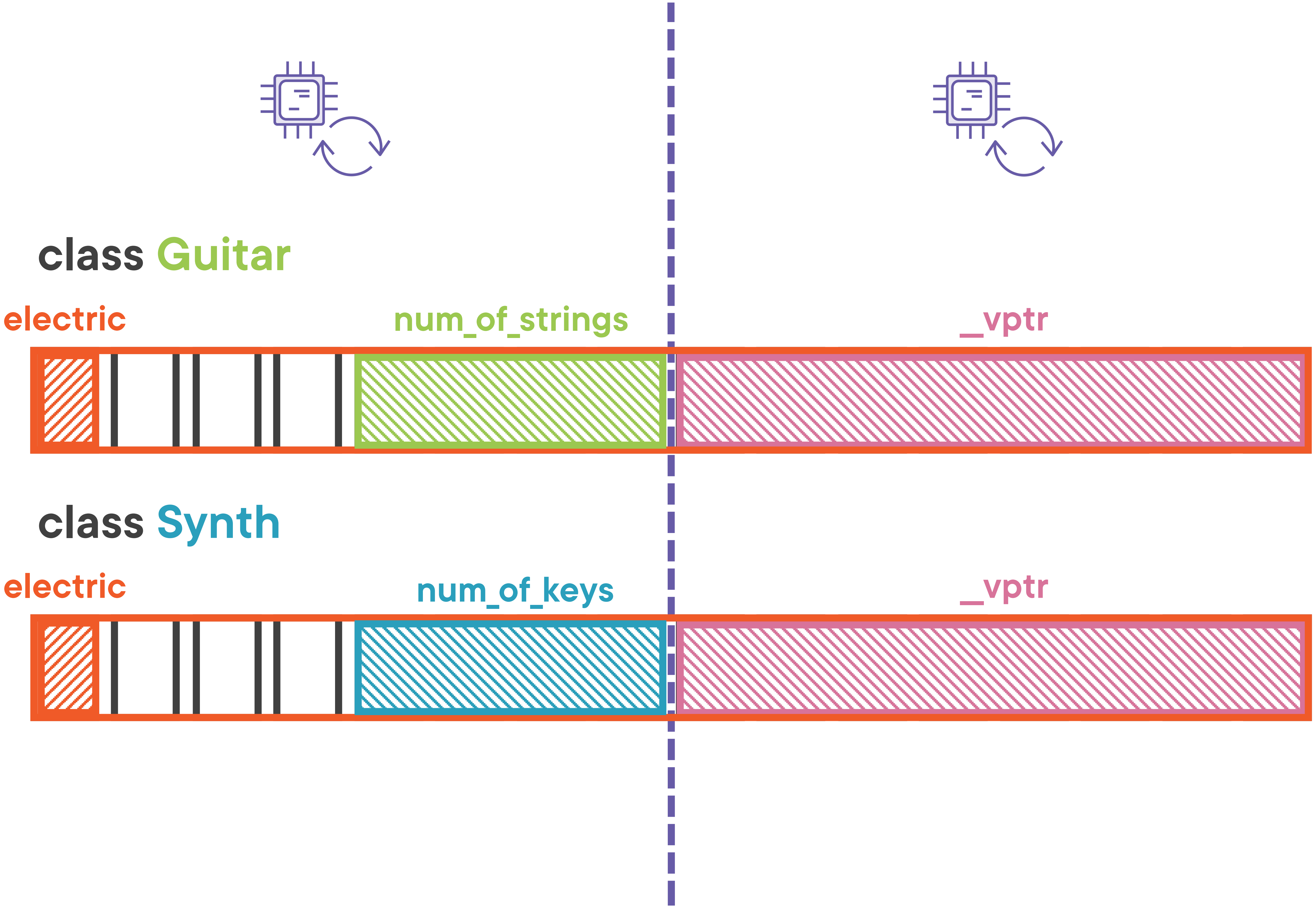
Data Structure Alignment



Data Structure Alignment



Data Structure Alignment



Value Categories in C++

Value Categories

lvalues

Value Categories

lvalues

prvalues

Value Categories

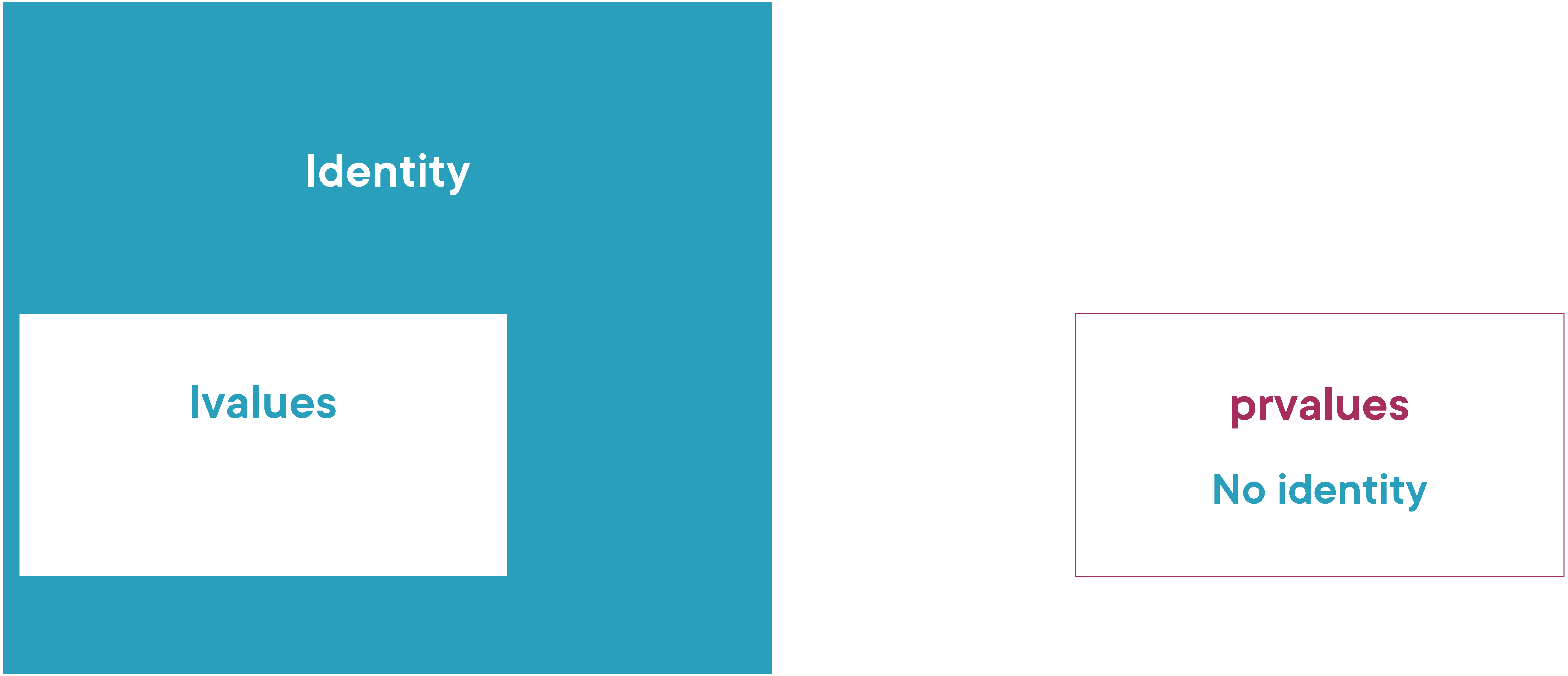
```
int a  =  5 ;
```

lvalues

prvalues

Value Categories

```
int a  =  5 ;
```



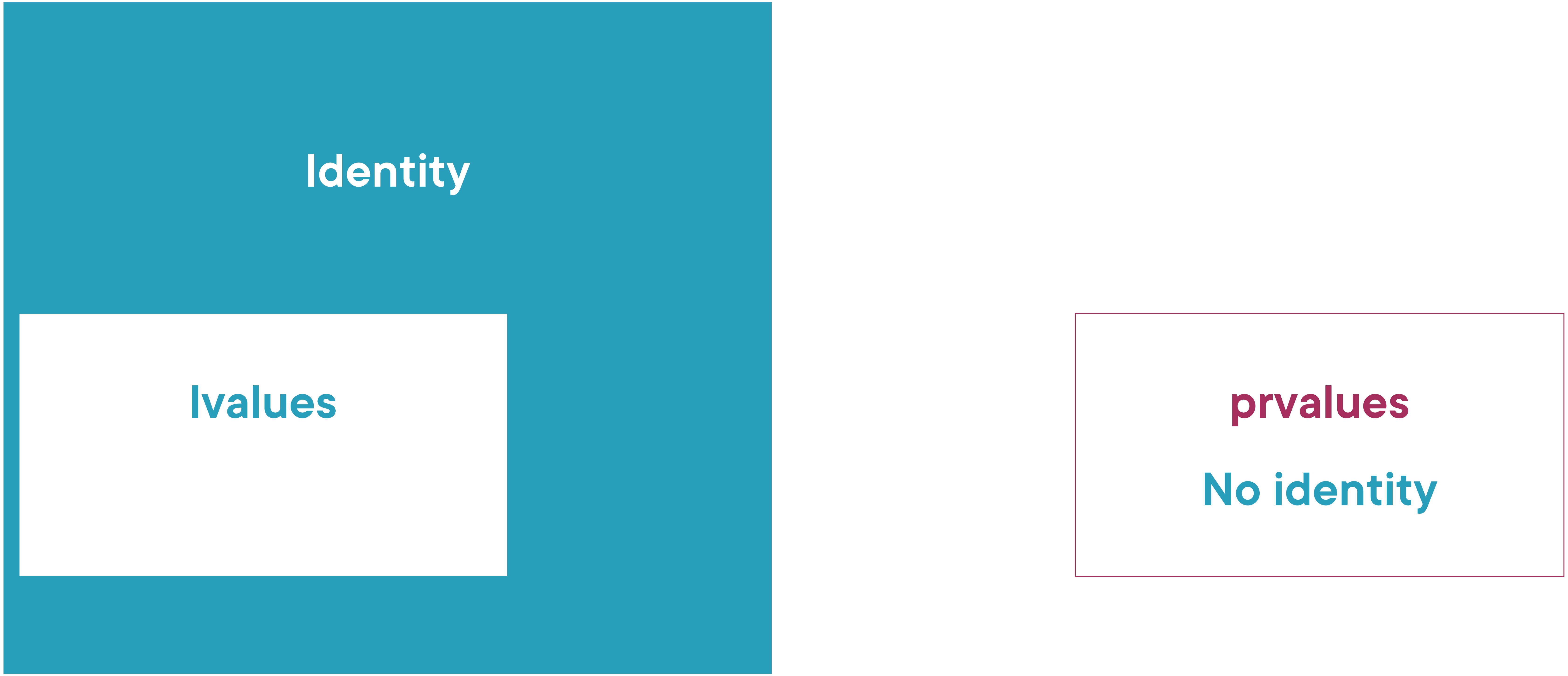
Identity

lvalues

prvalues

No identity

Value Categories



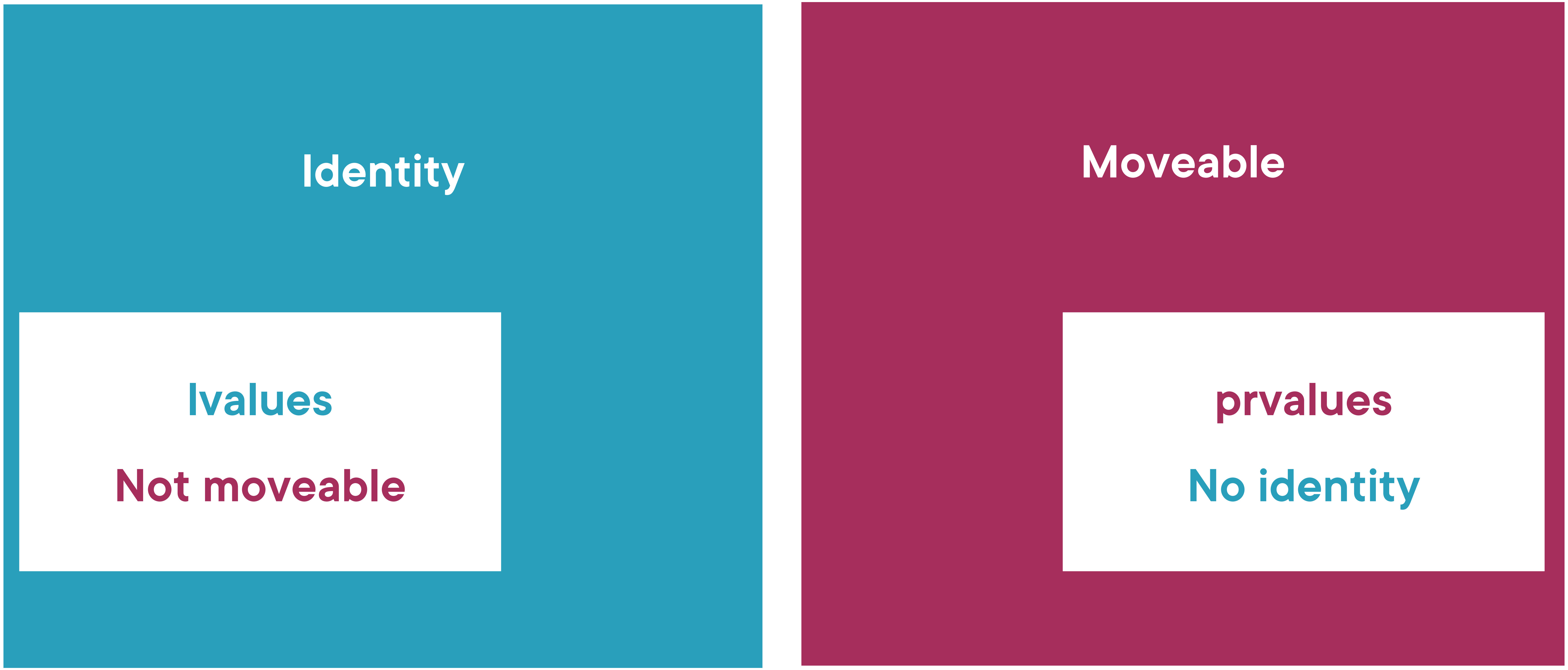
Identity

lvalues

prvalues

No identity

Value Categories



Identity

lvalues

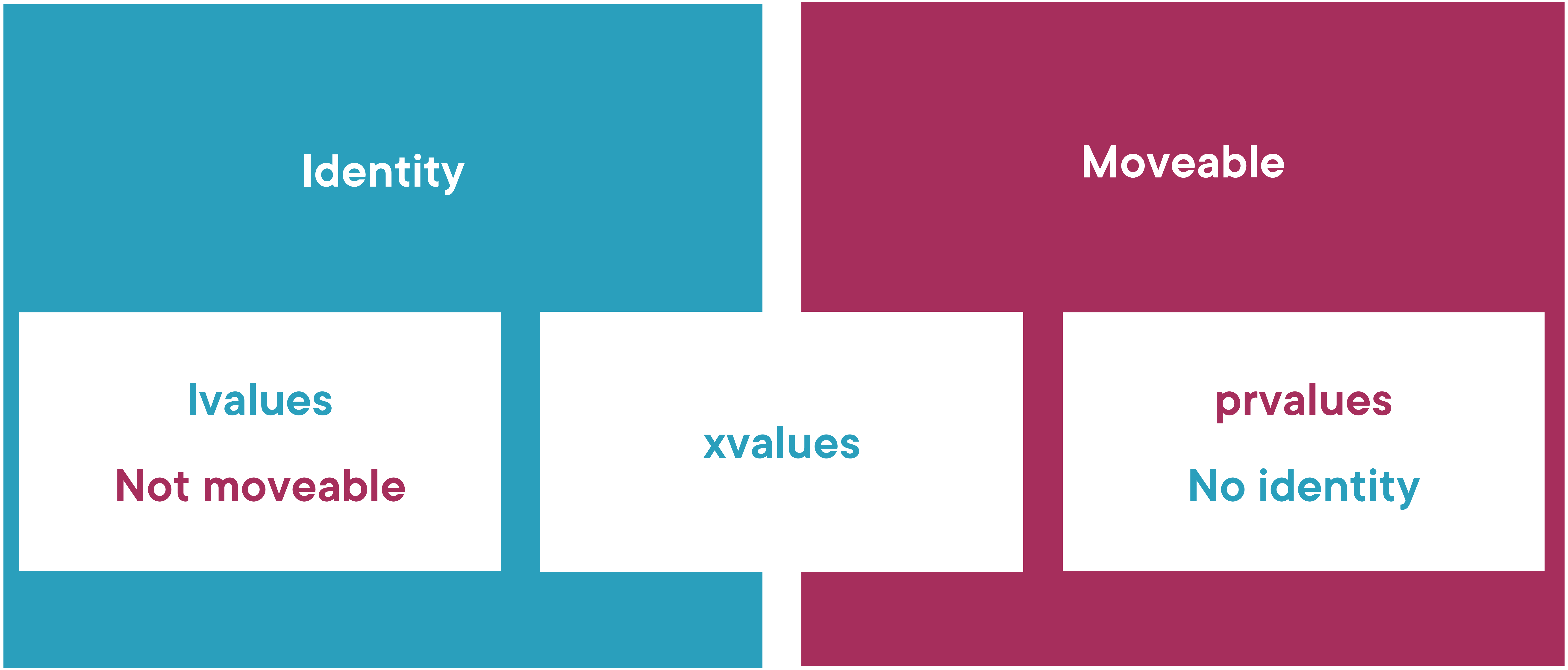
Not moveable

Moveable

prvalues

No identity

Value Categories



Identity

Moveable

lvalues

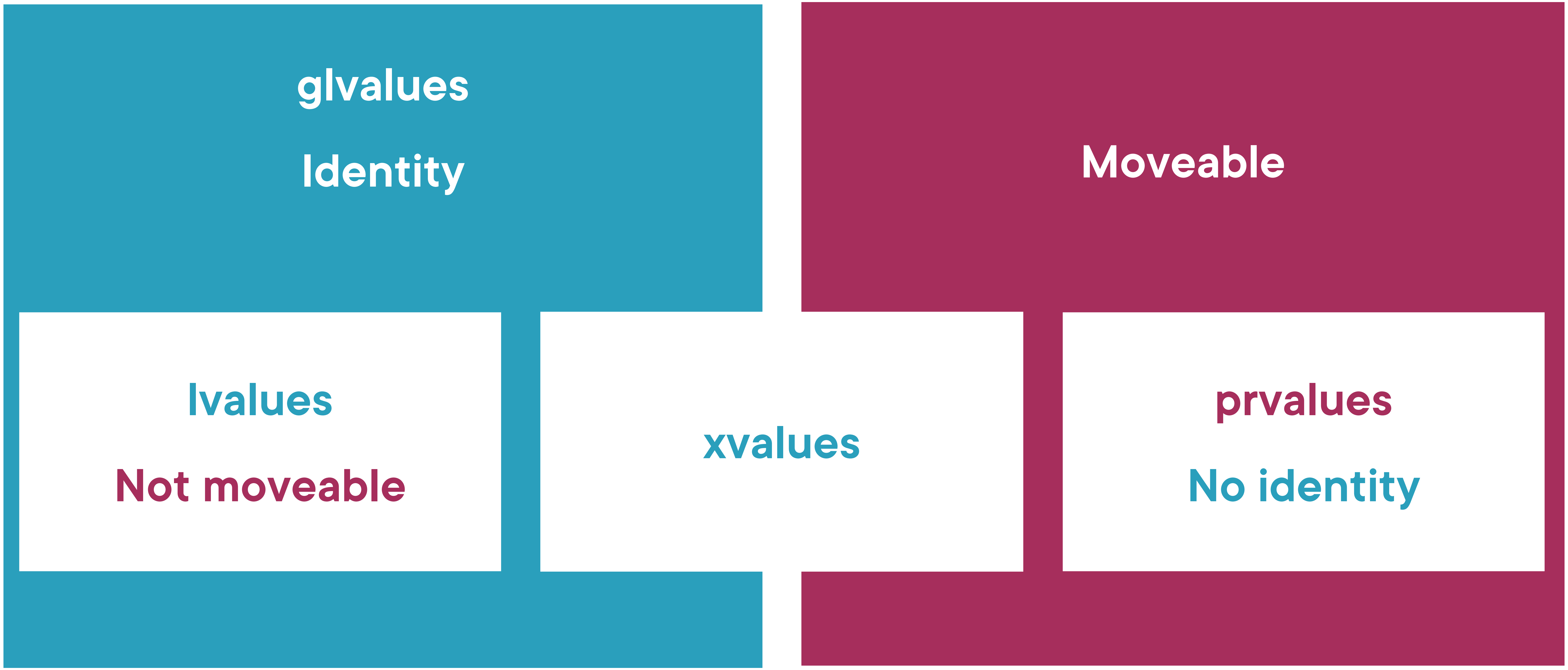
Not moveable

xvalues

prvalues

No identity

Value Categories



glvalues

Identity

Moveable

lvalues

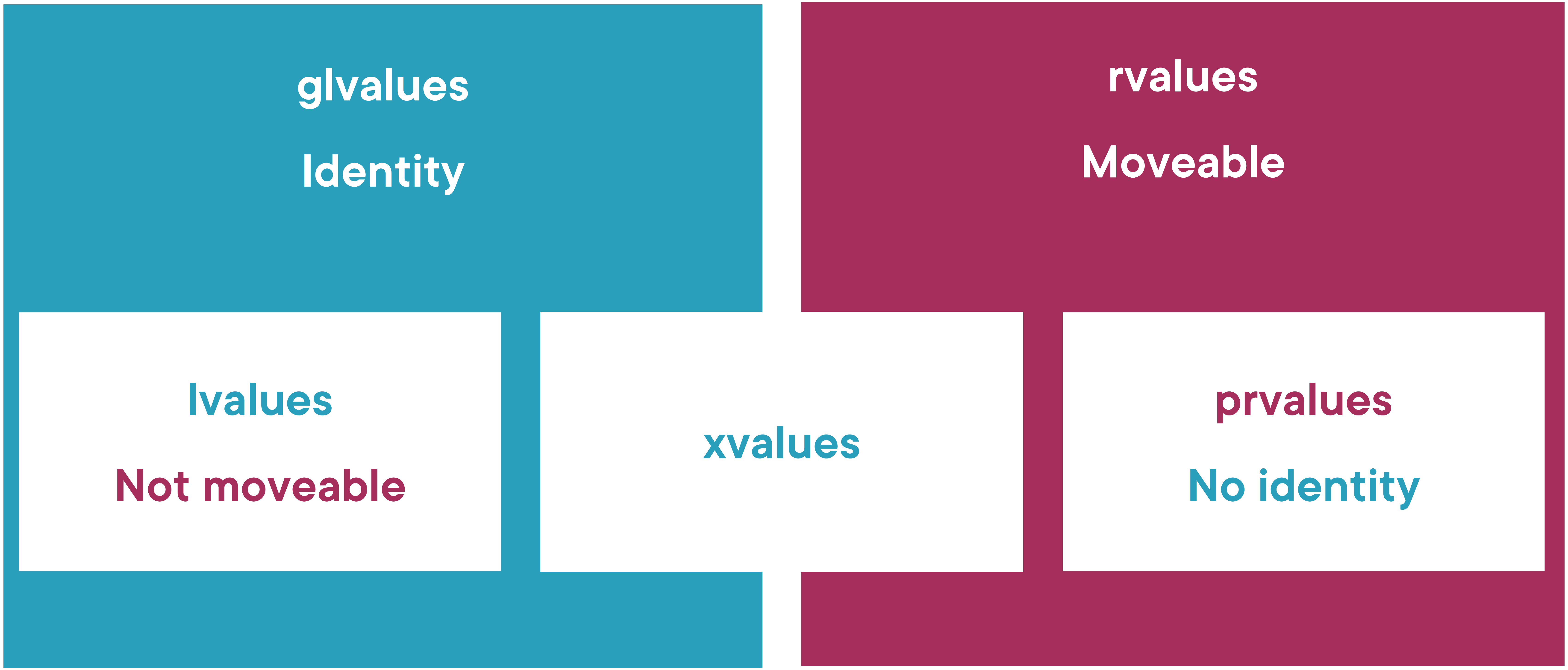
Not moveable

xvalues

prvalues

No identity

Value Categories



glvalues

Identity

rvalues

Moveable

lvalues

Not moveable

xvalues

prvalues

No identity

Up Next:
Abstracting Memory Management in Modern C++