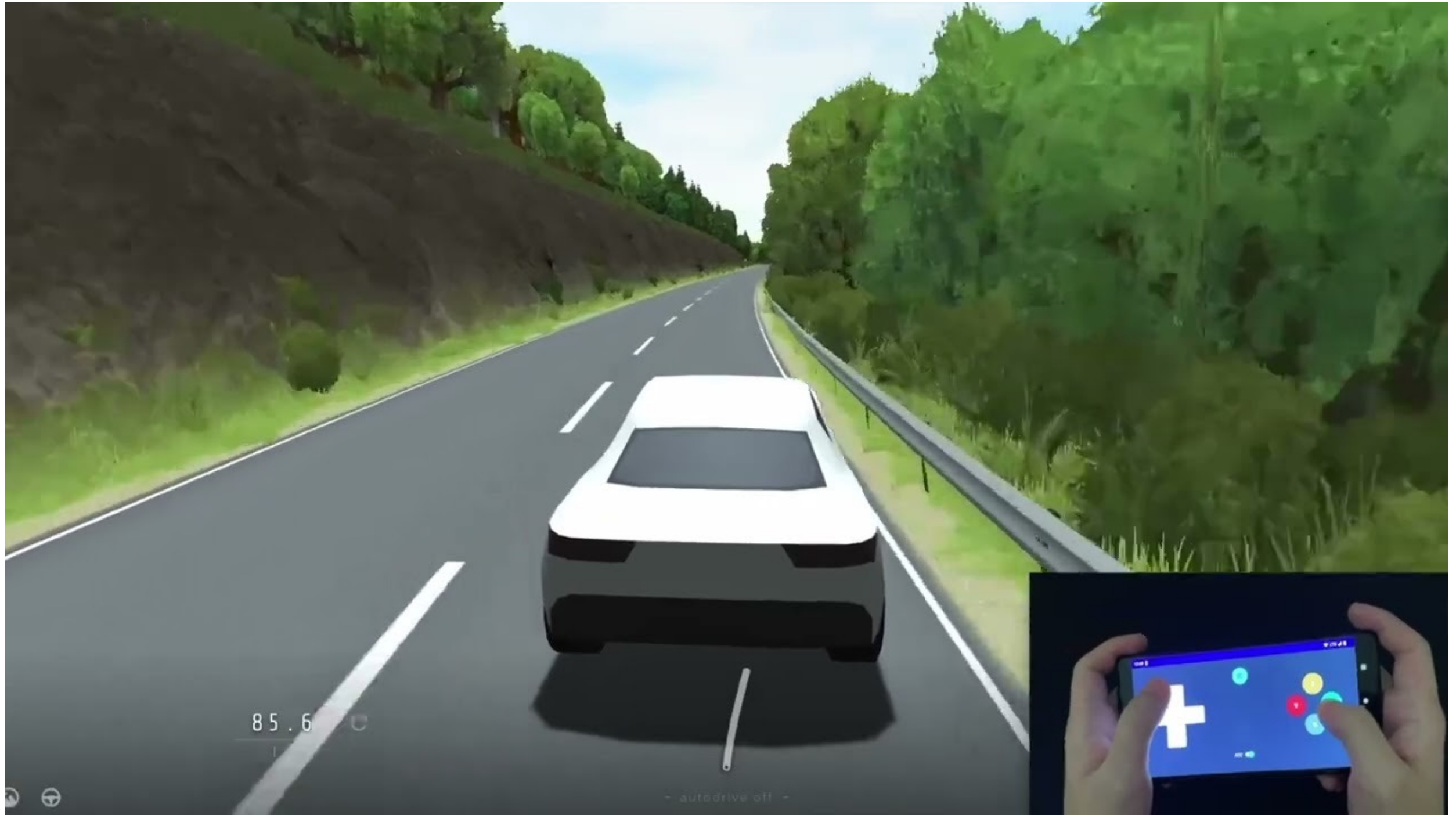




A Library for building Multi-device Applications

12조 미디어프로젝트 최종 발표



<https://youtu.be/9F0X0k7iHgk>

목차

1. 서론
2. 설계
 - 2.1. 라이브러리 아키텍처
 - 2.2. ZAPP (Zap Protocol)
3. 구현 및 배포
4. 예시 애플리케이션
5. 결론

1. 서론

- 한 명의 사용자가 보유한 스마트 기기 수가 꾸준히 증가함에 따라 기기간 유기적 결합이 중요해짐.
- 모바일 기기에는 동작 센서, 생체 센서, 터치스크린 등 유용한 장치가 다수 탑재되어 있음.
- 하지만, 장치들이 단일 기기에 종속되어 있어 활용이 어려움.

1. 서론

Zap: 멀티 디바이스 앱을 위한 프로그래밍 라이브러리



원격 리소스
추상화

크로스 플랫폼
사용자 경험 실현

1. 서론

Zap: 멀티 디바이스 앱을 위한 프로그래밍 라이브러리



원격 리소스
추상화

애플리케이션 계층에 추상화된
인터페이스를 제공함으로써 로컬
기기의 리소스에 접근하듯이 원격
기기의 리소스에 접근할 수 있어야 한다.

1. 서론

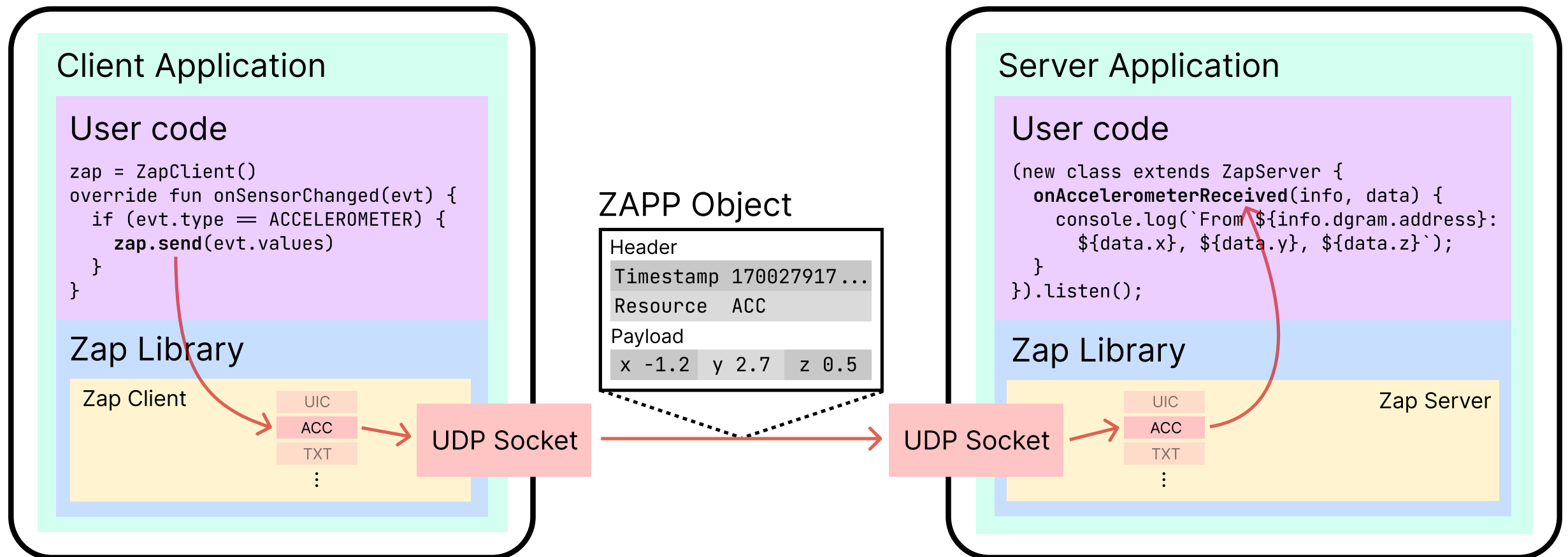
Zap: 멀티 디바이스 앱을 위한 프로그래밍 라이브러리

PC, TV, 키오스크와 같은 기기에는
모바일 기기에 탑재된 것과 같은 장치가
없으므로, 크로스 플랫폼을 지원해
확장된 멀티 디바이스 사용자 경험을
실현해야 한다.

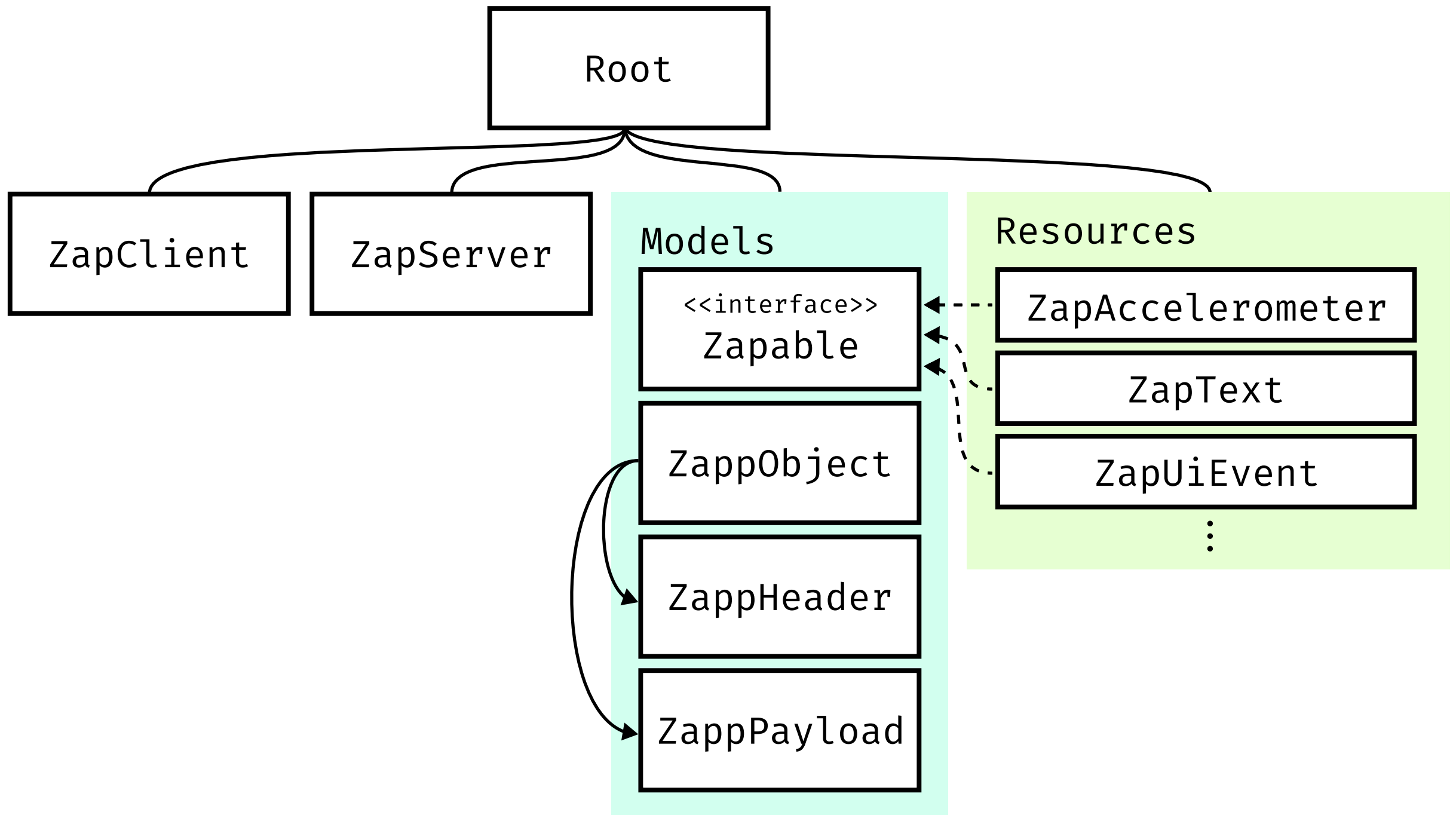


**크로스 플랫폼
사용자 경험 실현**

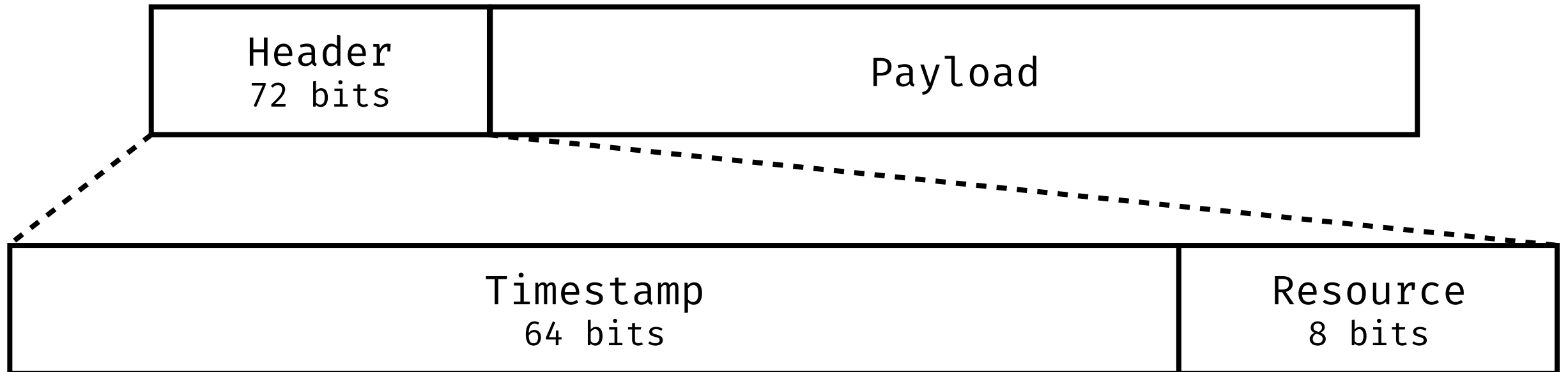
2. 설계



2.1. 설계: 라이브러리 아키텍처

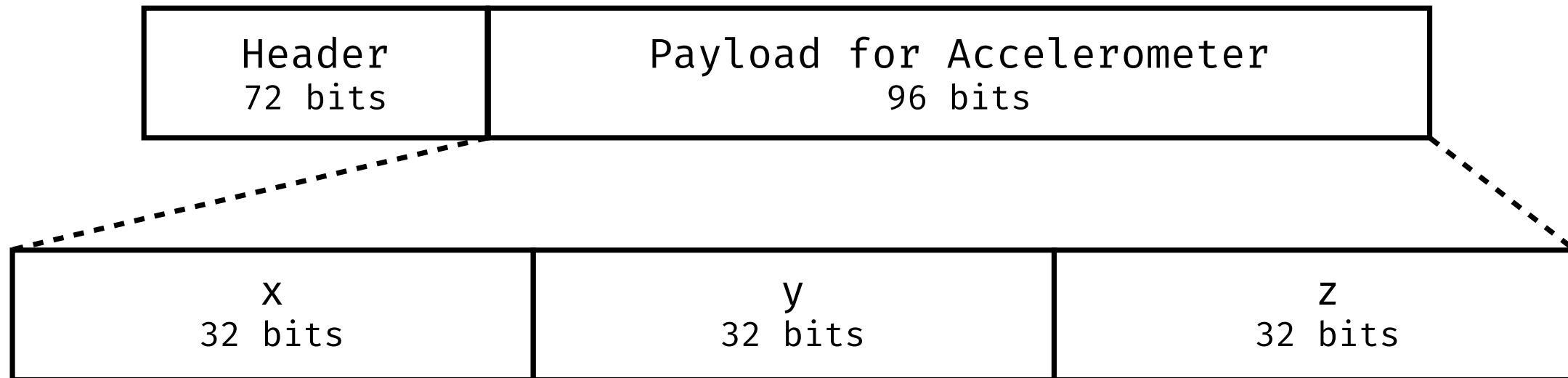


2.2. 설계: ZAPP (Zap Protocol)

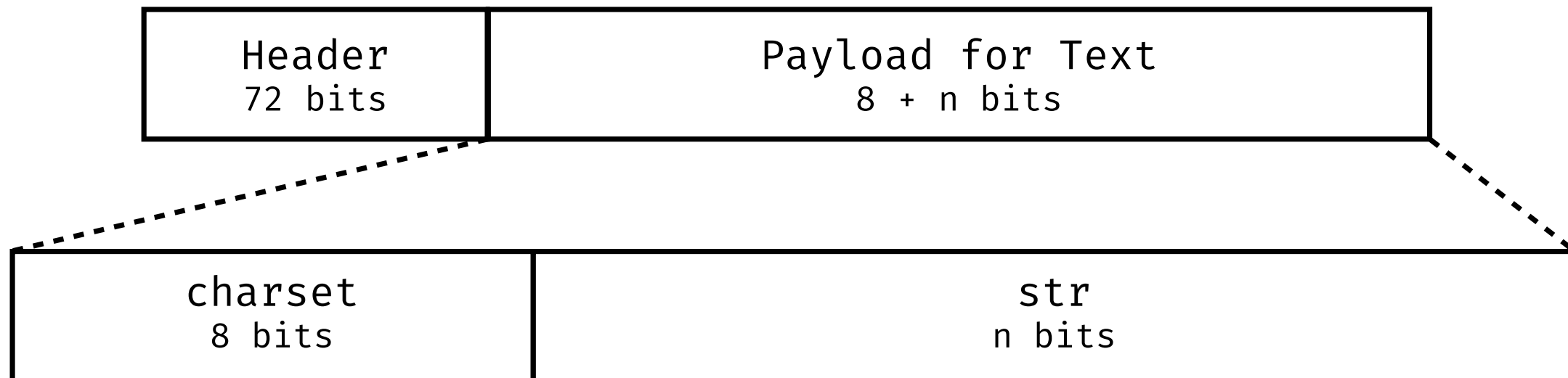


- 다양한 플랫폼간 데이터를 주고 받기 위한 프로토콜.
- UDP 위에 정의하여 높은 성능을 보장.
- 하나의 ZAPP Object는 헤더와 페이로드로 구성.
- 페이로드의 형식은 리소스에 따라 다르게 구성.

2.2. 설계: ZAPP (Zap Protocol)



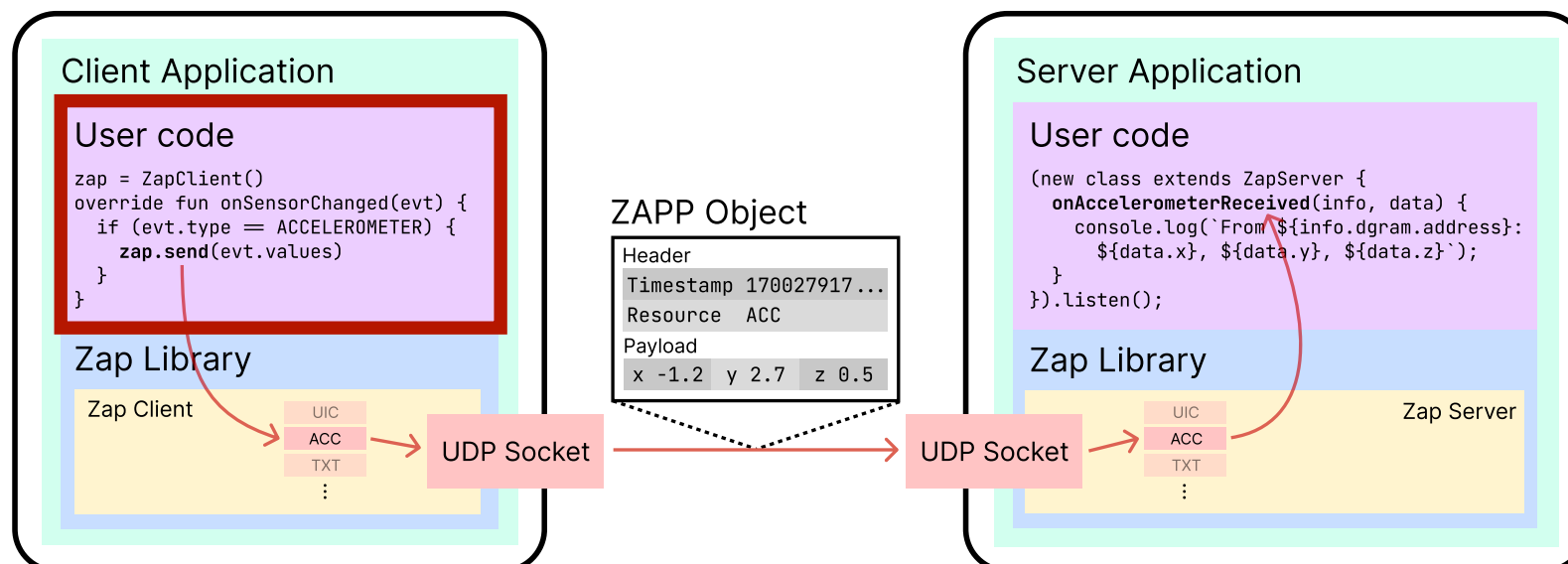
가속도계 데이터의 페이로드 형식



텍스트 데이터의 페이로드 형식

3. 구현 및 배포

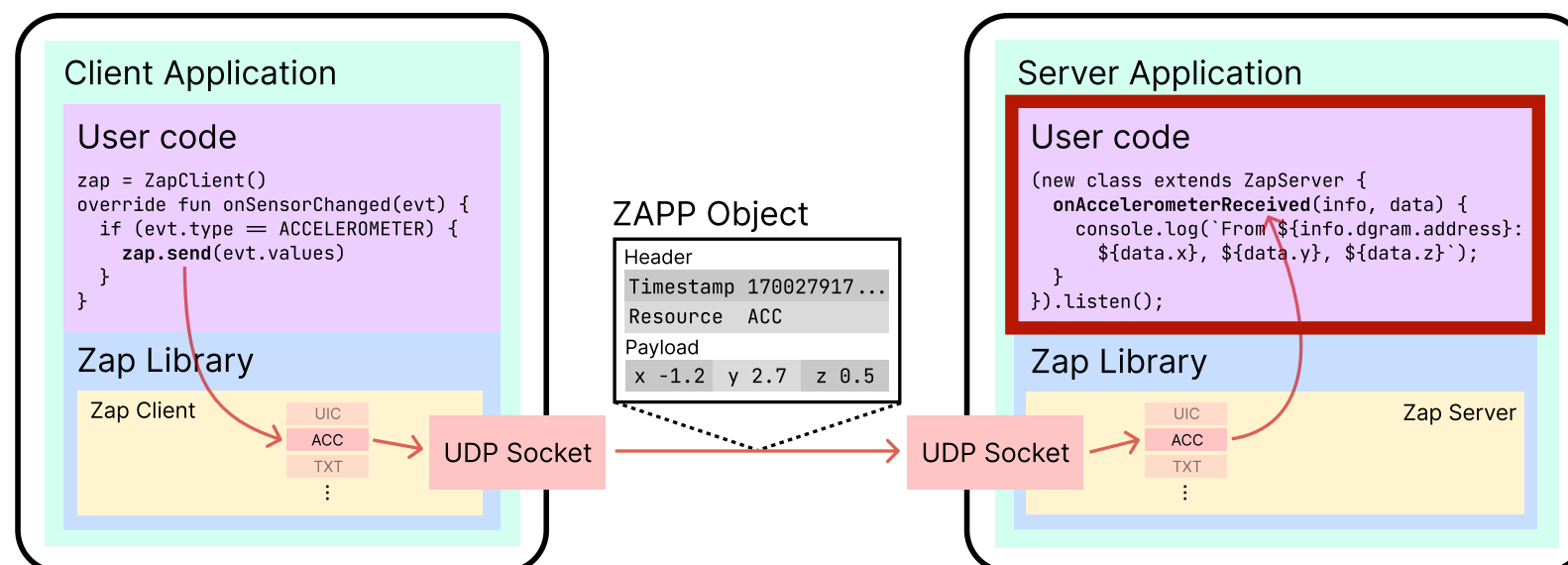
```
class MainActivity: AppCompatActivity(), SensorEventListener {  
    private lateinit var zap: ZapClient  
  
    override fun onCreate(state: Bundle?) {  
        zap = ZapClient(InetAddress.getByName(...))  
    }  
  
    override fun onSensorChanged(event: SensorEvent) {  
        if (event.sensor.type == Sensor.TYPE_ACCELEROMETER) {  
            val (x, y, z) = event.values  
            zap.send(ZapAccelerometer(x, y, z))  
        }  
    }  
}
```



3. 구현 및 배포

```
import { ZapServer } from 'zap-lib-js';

(new class extends ZapServer {
  onAccelerometerReceived(
    info: MetaInfo,
    data: ZapAccelerometer,
  ) {
    console.log(`Data received:
      ${data.x}, ${data.y}, ${data.z}`);
  }
}).listen();
```



3. 구현 및 배포

- 가속도 센서 (ZapAccelerometer)
- 중력 센서 (ZapGravity)
- 자이로스코프 센서 (ZapGyroscope)
- 조도 센서 (ZapIlluminance)
- 자기장 센서 (ZapMagneticField)
- 지리적 위치 (ZapGeoPoint)
- UI 이벤트 (ZapUiEvent)
- 텍스트 (ZapText)

3. 구현 및 배포

[illegible]

| | | | |
|---|---------------------------------------|--|----------------|
| Z | Zap | | Q Search |
| Introduction | Getting Started | Architectures | Specifications |
| Implementations | GitHub ↗ | | |
| Specifications | | | |
| Client and Server | | | |
| Models | | | |
| Resources | | | |
| Client and Server | | | |
| ZapClient | | | |
| A client sends data to server. | | | |
| type | signature | description | |
| property | <code>server_address</code> | An IP address of the device running <code>ZapServer</code> . | |
| function | <code>send(obj: Zapable): void</code> | Send given <code>Zapable</code> object to the server. <code>obj</code> : An object to send. | |
| function | <code>stop(): void</code> | Close the socket. | |
| ZapServer | | | |
| A server receives data from client. The 'open function' refers to a callback function that users SHOULD override when declaring a <code>ZapServer</code> object to define its behavior. Refer to the Resources section for information on the parameters of each open function. | | | |
| type | signature | description | |
| function | <code>listen(port: int): void</code> | Start listening the transmitted data from clients on the given port. <code>port</code> : A port number for receiving data (default: 65500). | |
| function | <code>stop(): void</code> | Stop listening to clients. | |

- 설계와 API 명세를 문서화해 웹으로 공개.
- zap-lib.github.io

3. 구현 및 배포

- Node.js 구현체는 npm 레지스트리로, Kotlin 구현체는 JitPack으로 배포.
- Apache License 2.0 라이선싱 및 오픈소스 공개.
(github.com/zap-lib)
- GitHub Action 워크플로우를 작성해 CI 구축.

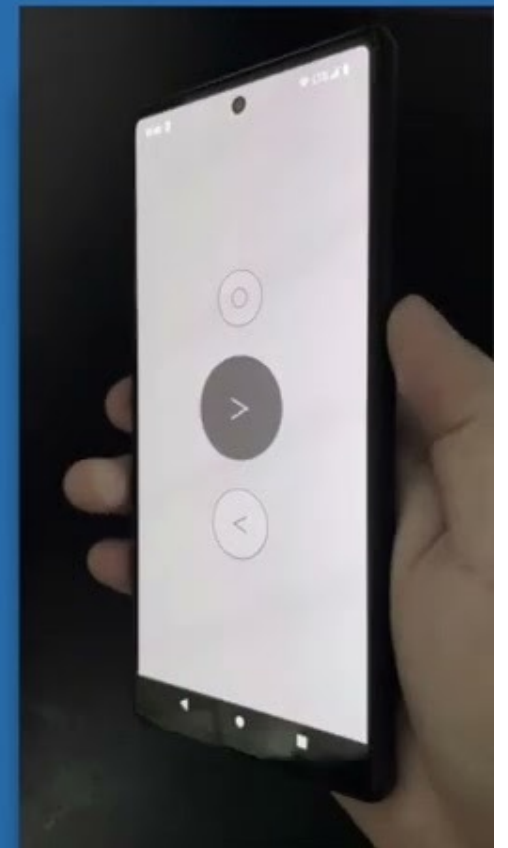
4. 예시 애플리케이션



<https://youtu.be/cJLLqGxvF3o>

4. 예시 애플리케이션

Slide 1



<https://youtu.be/AYFn2OB8KvY>

4. 예시 애플리케이션

Try writing by hand on the client device.

世界人權宣言



<https://youtu.be/53asTThWCFM>

5. 결론

- 플랫폼 독립적인 원격 리소스 공유 API와 프로토콜을 제공.
- 모바일-모바일 결합이 아닌 크로스 플랫폼을 지원하는 공개된 솔루션을 제시함으로써 기존 멀티 디바이스 연구와 차별성을 확보.
- 단일 기기의 한계를 극복해 멀티 디바이스 프로그램의 무궁무진한 가능성을 실현할 수 있을 것으로 기대.

참고문헌

- Naser AlDuaij and Jason Nieh. 2021. Tap: an app framework for dynamically composable mobile systems. In *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys 2021)*. Virtual, 336–349.
- Naser AlDuaij, Alexander Van't Hof, and Jason Nieh. 2019. Heterogeneous Multi-Mobile Computing. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys 2019)*. Seoul, South Korea, 494–507.
- Sangeun Oh, Ahyeon Kim, Sunjae Lee, Kilho Lee, Dae R. Jeong, Steven Y. Ko, and Insik Shin. 2019. FLUID: Flexible User Interface Distribution for Ubiquitous Multi-Device Interaction. In *Proceedings of the 25th Annual International Conference on Mobile Computing and Networking (MobiCom 2019)*. Los Cabo, Mexico, 1–16.
- AirConsole, www.airconsole.com.
- Swip.js, github.com/paulsonnentag/swip.