

Chapter 09 모두의 숙소 앱 만들기

완성 화면

The image shows a mobile application interface for RoomOfAll. At the top left is the logo 'RoomOfAll' with a heart icon. At the top right are links for '회원가입' (Sign Up) and '로그인' (Log In). The main content area features a large search bar with the placeholder '모두의 숙소에서 숙소를 검색하세요.' (Search for accommodations from everyone's place). Below the search bar is a descriptive text: '흔지하는 여행에 적합한 개인실부터 여벗이 함께 하는 여행에 좋은 '공간전체' 숙소까지, 모두의 숙소에 다 있습니다.' (From personal rooms suitable for solo travel to rooms where roommates are welcome, find all kinds of accommodations here). To the right of the text is a photograph of a boat's deck floating on turquoise water. On the left side of the search bar is a form with input fields: '위치' (Location) with '근처 추천 장소' (Nearby recommended places), '체크인' (Check-in) and '체크아웃' (Check-out) with '날짜 입력' (Date input), and '성인' (Adults) with '2' and '어린이' (Children) with '0'. A red '검색' (Search) button is at the bottom of the form. Below this section are two large photographs of travel destinations: a waterfall cascading down a red rock cliff and a modern interior room with a brown sofa and a large window overlooking a city skyline.

The screenshot shows two main sections. On the left, a search interface with fields for location, check-in/check-out dates, and guest count, followed by a red '검색' (Search) button. On the right, a listing detail page featuring a large image of a terrace at sunset, a 5-star rating, and a detailed description in Korean.

모두의 숙소에서 숙소를 검색하세요.

혼자하는 여행에 적합한 개인실부터 여럿이 함께 하는 여행에 좋은 '공간전체' 숙소까지, 모두의 숙소에 다 있습니다.

위치
근처 추천 장소

체크인
날짜 입력

체크아웃
날짜 입력

성인
2

여린이
0

검색

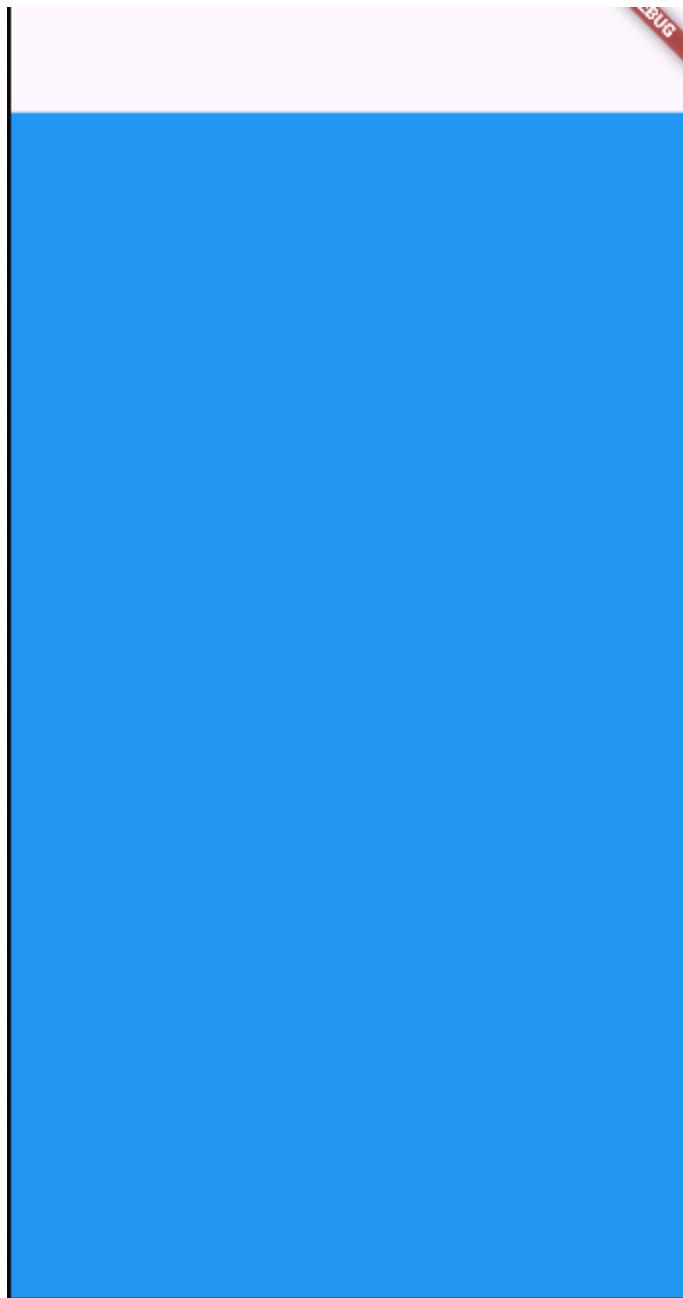
★★★★★
깔끔하고 다 갖춰져 있어서 좋았어요:) 위치도 완전 좋아용 다들 여기 살고싶다구ㅋㅋㅋㅋㅋ 화장실도 3개예요!!! 5명이서 씻는 것도 전혀 불…

데어
한국

1. double.infinity

💡 `double.infinity`는 Dart 언어에서 사용되는 특수한 상수로, 양의 무한대를 의미한다. Flutter에서 `double.infinity`는 주로 위젯의 크기를 부모 위젯의 크기에 맞추기 위해 사용된다. 예를 들어, 특정 위젯의 너비나 높이를 `double.infinity`로 설정하면, 해당 위젯은 가능한 한 최대 크기로 확장된다. 이는 부모 위젯의 제약에 따라 크기가 결정되며, 주어진 공간을 모두 사용하고자 할 때 유용하다.

```
body: Center(
  child: Container(
    width: double.infinity,
    height: double.infinity,
    color: Colors.blue,
  ), // Container
), // Center
```



double.infinity 를 사용하면 부모 위젯의 크기 만큼 공간을 차지한다.

2. MediaQuery 클래스

💡 MediaQuery는 화면의 크기, 해상도, 밝기 등의 정보에 접근할 수 있게 해주는 클래스다. 이를 통해 Flutter 애플리케이션이 실행되는 기기의 화면 크기나 화면의 다양한 특성에 따라 UI를 동적으로 조정할 수 있다. 화면 크기와 해상도, 화면 방향 등을 조정할 수 있다.

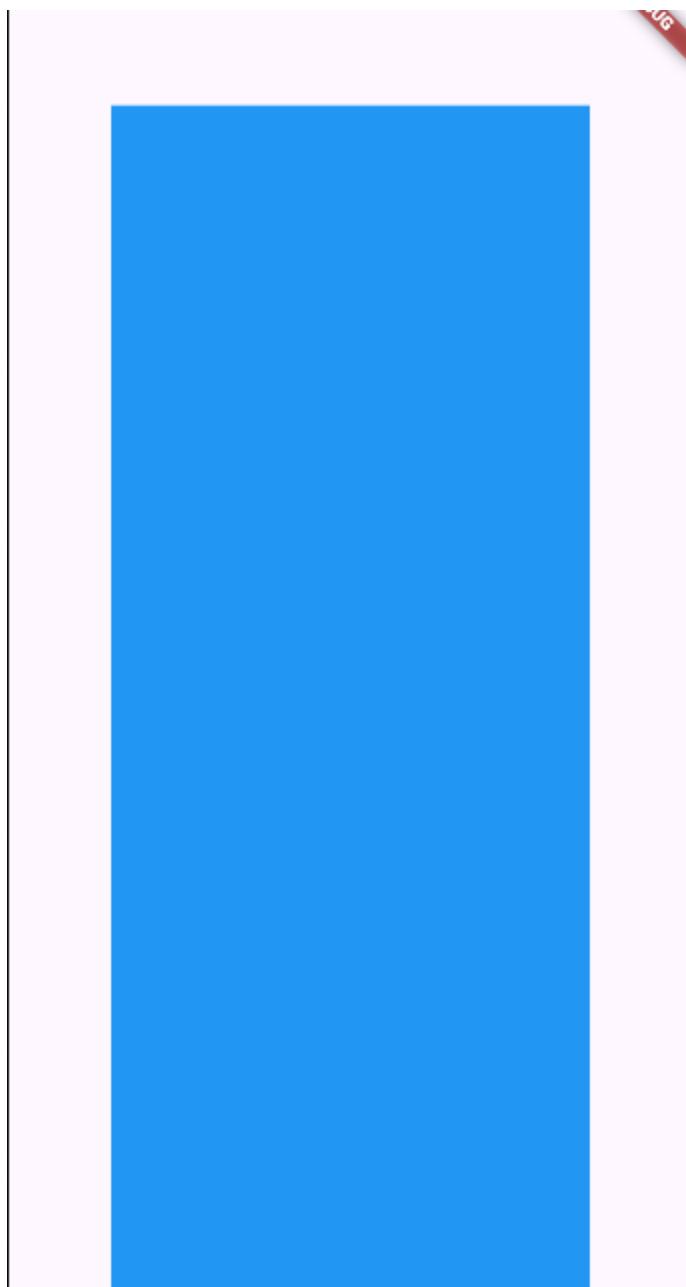
주요 속성은 다음과 같다.

1. size: 화면의 너비와 높이 정보를 담고 있는 Size 객체.
2. devicePixelRatio: 논리 픽셀과 실제 픽셀 간의 비율. 고해상도 디스플레이에서 유용하게 쓸 수 있다.
3. orientation: 화면이 세로(Orientation.portrait)인지 가로(Orientation.landscape)인지를 나타낸다.
4. padding: 안전 영역에 대한 정보(예: 상태 표시줄, 노치 영역 등을 고려한 패딩).
5. textScaleFactor: 텍스트의 배율을 나타내며, 사용자가 텍스트 크기를 조정한 경우에 유용하다.

```
body: Center(
  child: Container(
    width: MediaQuery.of(context).size.width * 0.7,
    height: double.infinity,
    color: Colors.blue,
  ), // Container
) // Center
```

MediaQuery.of(context).size 를 통해 앱의 가로 길이를 디바이스의 가로 길이의 0.7 비율로 설정한다.

이는 다양한 화면 크기에서 동일한 비율로 UI 요소를 표시하여 일관된 사용자 경험을 제공하기 위함이다. 예를 들어, 큰 화면에서는 UI 요소가 너무 넓어지지 않도록 하고, 작은 화면에서는 충분한 너비를 확보하도록 조정할 수 있다.



3. Align 위젯

💡 Align 위젯은 Flutter에서 자식 위젯의 정렬을 조정하는 데 사용된다. Align 위젯은 alignment 속성을 통해 자식 위젯이 부모 위젯 내에서 어디에 위치할지를 결정한다.

Align 위젯의 주요 속성은 다음과 같다.

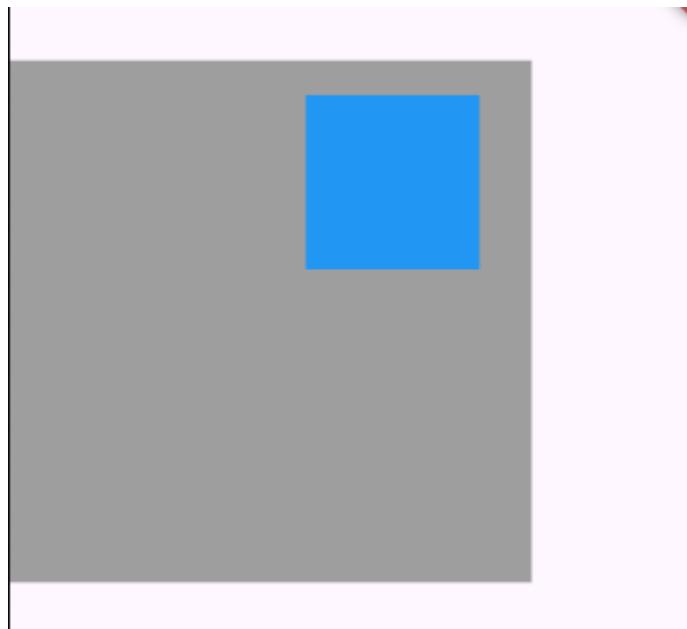
1. alignment: 자식 위젯을 부모의 어떤 위치에 배치할지를 정의한다. Alignment 객체를 사용하여 정의되며, 기본적인 값으로는 Alignment.center, Alignment.topLeft, Alignment.bottomRight 등이 있다.
2. widthFactor: 부모 위젯의 너비를 자식 위젯의 너비에 맞추는 요소. 값이 null이면, 부모 위젯은 가능한 최대 너비를 차지하게 된다. 값이 1이면, 부모 위젯의 너비는 자식 위젯의 너비와 동일하게 설정된다.
3. heightFactor: widthFactor와 유사하게, 부모 위젯의 높이를 자식 위젯의 높이에 맞추는 요소.

```
body: Container(
  color: Colors.grey,
  height: 300,
  width: 300,
  child: Align(
    alignment: Alignment.centerLeft,
    child: Container(
      color: Colors.blue,
      width: 100,
      height: 100,
    ), // Container
  ) // Align
```



Alignment의 기본 값으로 배치할 수 있다.

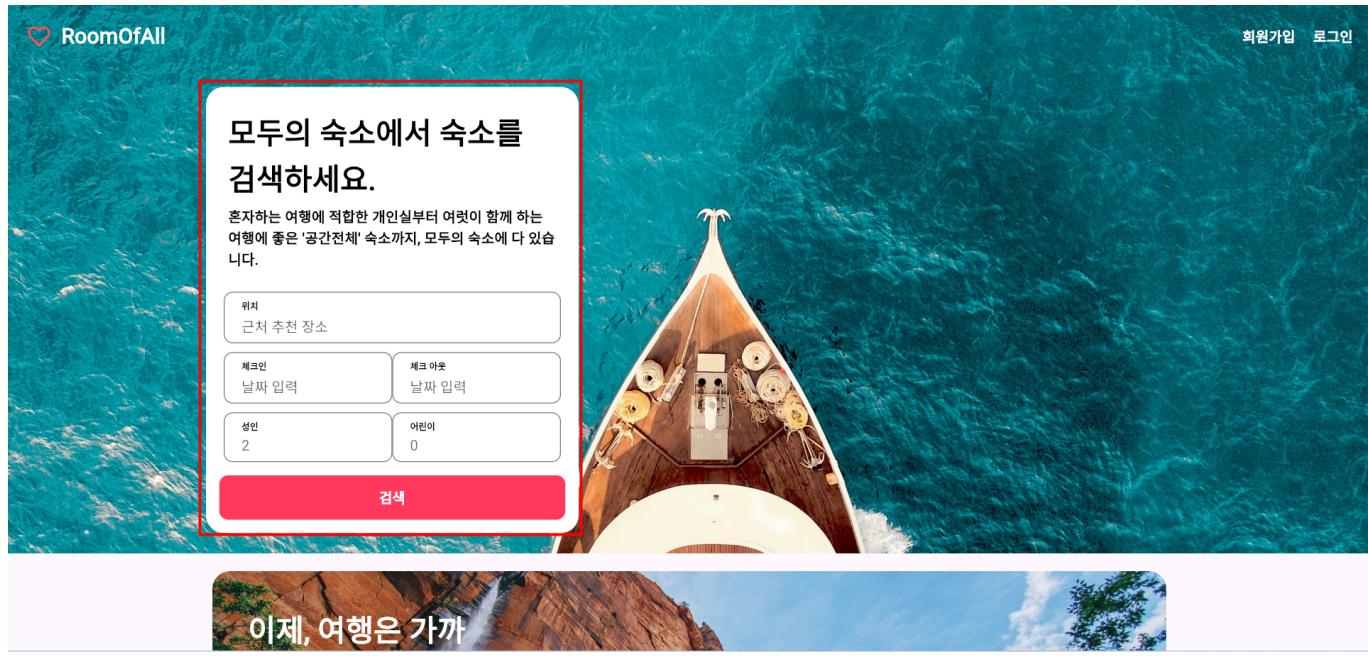
```
width: 300,  
child: Align(  
  alignment: Alignment(0.7,-0.8),  
  child: Container(  
    color: Colors.blue,  
    width: 100,  
    height: 100,  
  ), // Container  
, // Align  
) // Container
```



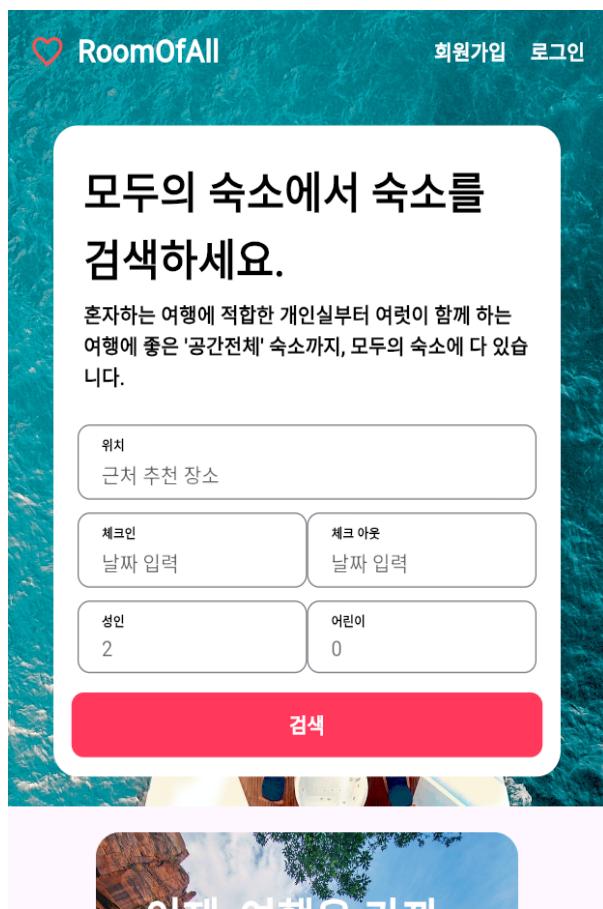
자식 위젯의 위치를 직접 지정할 수도 있다.

```
padding: const EdgeInsets.only(top: gap_m),  
child: Align(  
  alignment:  
    screenWidth < 520 ? Alignment(0, 0) : Alignment(-0.6, 0),  
  child: Container(  
    width: 420,  
    decoration: BoxDecoration(  
      color: Colors.white,
```

삼항연산자를 사용해 디바이스의 가로 길이가 변경됐을 때 자식 위젯의 위치를 동적으로 변경할 수 있다.



사이즈가 클 때는 부모 위젯의 왼쪽에 위치하지만 사이즈가 줄어들면 부모 위젯의 가운데에 위치하도록 한다.



4. Wrap 위젯

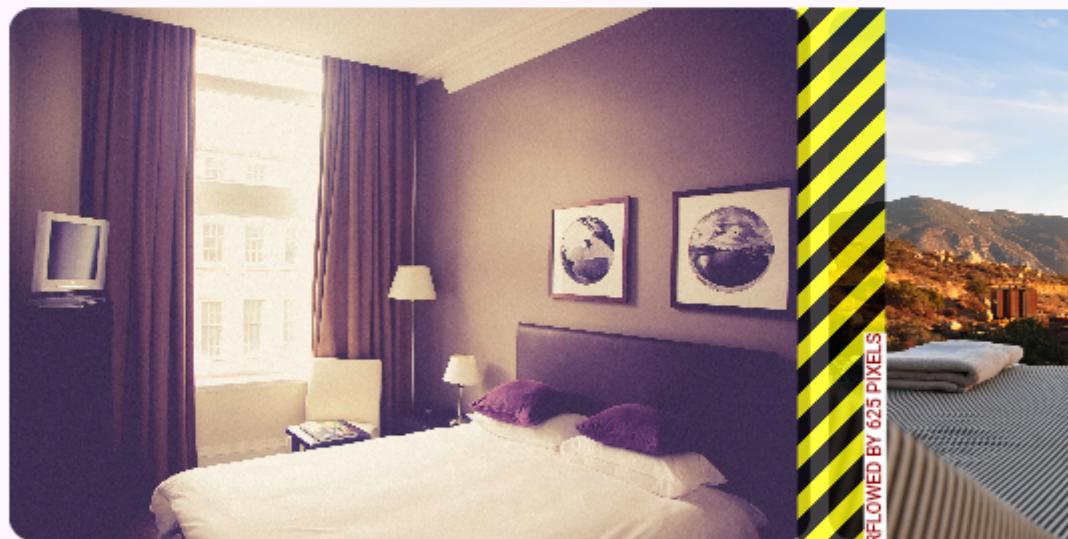
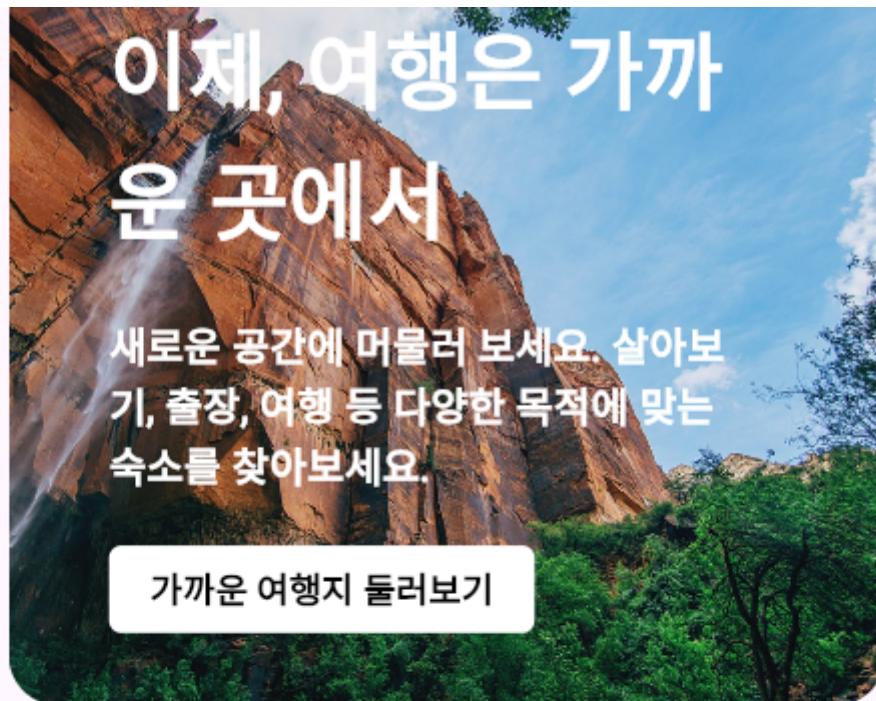
💡 Wrap 위젯은 Flutter에서 자식 위젯들을 자동으로 다음 줄로 넘겨 배치하는 데 사용되는 레이아웃 위젯이다. 이는 자식 위젯들이 수평 또는 수직 방향으로 오버플로우할 때 유용하며, 그리드 레이아웃과 유사한 방식으로 요소들을 배치할 수 있다.

Wrap 위젯은 Row와 Column의 기능을 결합한 형태로 볼 수 있으며, 자식 요소들이 지정된 방향으로 가득 차면 다음 줄이나 열로 이동하여 배치된다.

Wrap 위젯의 주요 속성은 다음과 같다.

1. direction: Wrap 위젯의 주 축 방향을 설정한다. 기본값은 Axis.horizontal로, 수평 방향으로 자식 위젯을 배치합니다. 수직 방향으로 배치하려면 Axis.vertical을 사용한다.
2. alignment: 주 축 방향에서 자식 위젯들의 정렬 방식을 결정한다. 예를 들어, WrapAlignment.start, WrapAlignment.center, WrapAlignment.end 등을 사용할 수 있다.
3. spacing: 주 축 방향에서 자식 위젯들 사이의 간격을 설정한다.
4. runAlignment: Wrap 위젯의 교차 축에서 줄 또는 열의 정렬 방식을 설정한다.
5. runSpacing: 교차 축에서 줄 또는 열 간의 간격을 설정한다.
6. children: Wrap 위젯의 자식 위젯을 담는다.

```
Row(  
  children: [  
    HomeBodyPopularItem(id:0),  
    SizedBox(width: 7.5),  
    HomeBodyPopularItem(id:1),  
    SizedBox(width: 7.5),  
    HomeBodyPopularItem(id:2),  
  ],  
), // Row
```



깔끔하고 다 갖춰져 있어서 좋았어요:) 위치도 완전 좋아용 다들 여기 살고 싶다구ㅋㅋㅋㅋㅋ 화장실도 3개예요!!! 5명이서 씻는 것도 전혀 불…



데어
한국



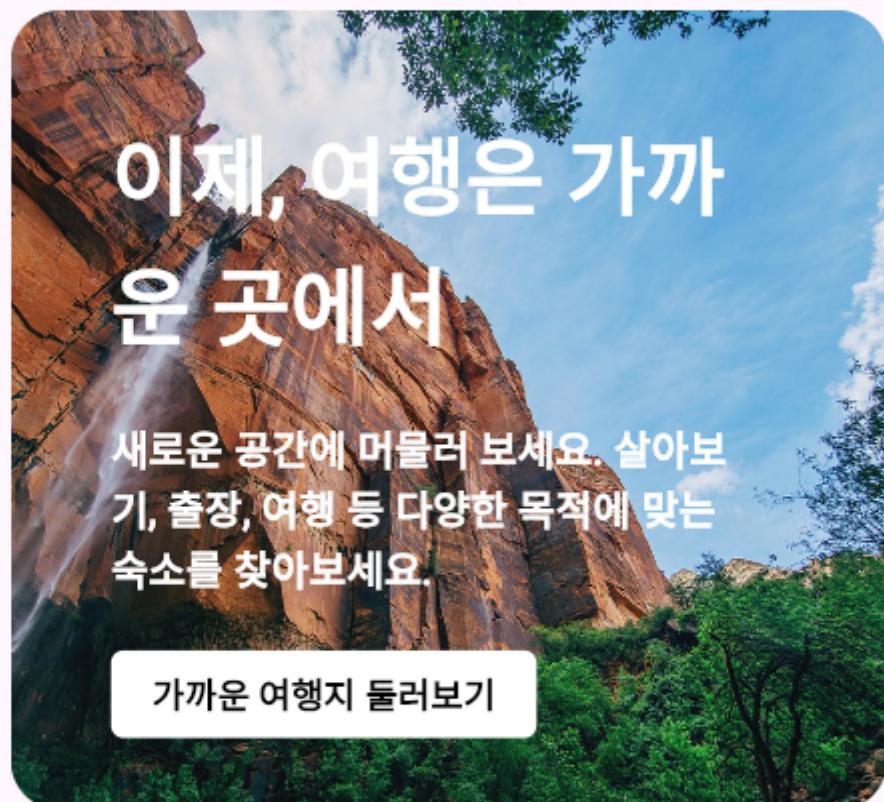
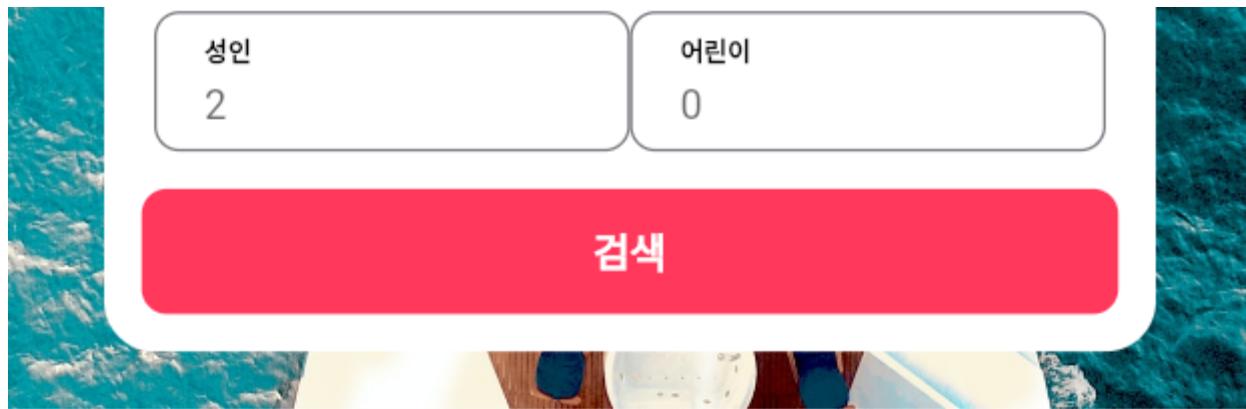
RIGHT OVERFLOWED BY 625 PIXELS



데어
한국

자식 위젯을 Row로 배치하면 가로 길이가 작아졌을 때 overflowed 오류가 발생하게 된다.

```
        SizedBox(),
      Wrap(
        children: [
          HomeBodyPopularItem(id:0),
          SizedBox(width: 7.5),
          HomeBodyPopularItem(id:1),
          SizedBox(width: 7.5),
          HomeBodyPopularItem(id:2),
        ],
      ), // Wrap
```



Wrap 위젯을 사용하면 사이즈가 줄어도 자동으로 교차 축 정렬이 된다.