ตอนที่ 2 การพัฒนาแอพพลิเคชั่นใน React Native

With พื้นฐานที่ครอบคลุมคุณสามารถเริ่มต้นการเพิ่มคุณสมบัติในการตอบสนองพื้นเมืองของคุณ
แอป. บทในส่วนนี้ครอบคลุมถึงสไตล์ การนำทาง แอนิเมชั่น และความสง่างาม
วิธีจัดการข้อมูลโดยใช้สถาปัตยกรรมข้อมูล (โดยเน้นที่ Redux)
บทที่ 4 และ 5 สอนวิธีใช้สไตล์แบบอินไลน์กับส่วนประกอบหรือ
ในสไตล์ชีตที่คอมโพเนนต์สามารถอ้างอิงได้ และเนื่องจาก React Native compo-
nents เป็นส่วนประกอบหลักของ UI ของแอปของคุณ บทที่ 4 ใช้เวลาพอสมควร
สอนสิ่งที่มีประโยชน์ที่คุณสามารถทำได้ด้วยองค์ประกอบมุมมอง บทที่ 5 สร้างขึ้นบน
ทักษะที่สอนในบทที่ 4 ครอบคลุมแง่มุมต่างๆ ของสไตล์ที่เจาะจงแพลตฟอร์ม
ตลอดจนเทคนิคขั้นสูงบางอย่าง รวมถึงการใช้ flexbox เพื่อให้ง่ายต่อการ
วางใบสมัคร
บทที่ 6 แสดงวิธีใช้ระบบนำทางสองระบบที่แนะนำและใช้มากที่สุด
ไลบรารี gation, React Navigation และ React Native Navigation เราเดินผ่าน
การสร้างเนวิเกเตอร์หลักสามประเภท—แท็บ สแต็ก และลิ้นชัก—และวิธีการ
ควบคุมสถานะการนำทาง
บทที่ 7 ครอบคลุมสี่สิ่งที่คุณต้องทำเพื่อสร้างแอนิเมชั่น สี่
ประเภทของส่วนประกอบที่เคลื่อนไหวได้ซึ่งมาพร้อมกับ Animated API วิธีสร้าง
ส่วนประกอบที่เคลื่อนไหวได้เอง และทักษะที่มีประโยชน์อื่นๆ อีกหลายอย่าง
ในบทที่ 8 เราสำรวจการจัดการข้อมูลด้วยสถาปัตยกรรมข้อมูล เพราะ Redux
เป็นวิธีที่ใช้กันอย่างแพร่หลายในการจัดการข้อมูลในระบบนิเวศ React คุณ
ใช้เพื่อสร้างแอปในขณะเดียวกันก็เรียนรู้ทักษะการจัดการข้อมูล เราแสดงวิธีการ
ใช้ Context API และวิธีใช้ Redux กับแอป React Native โดย
ใช้ตัวลดเพื่อคงสถานะ Redux และลบรายการออกจากแอพตัวอย่าง
นอกจากนี้เรายังครอบคลุมถึงวิธีการใช้ผู้ให้บริการเพื่อส่งสถานะทั่วโลกไปยังส่วนที่เหลือของแอป อย่างไร
เพื่อใช้ฟังก์ชันเชื่อมต่อเพื่อเข้าถึงแอปตัวอย่างจากคอมโพเนนต์ลูก
และวิธีการใช้การกระทำเพื่อเพิ่มฟังก์ชันการทำงาน
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

75

รู้เบื้องต้นเกี่ยวกับสไตล์

บทนี้ครอบคลุม

- จัดแต่งทรงผมโดยใช้ Javascript
- การใช้และการจัดรูปแบบ
- การใช้สไตล์เพื่อดูส่วนประกอบ
- การใช้สไตล์กับองค์ประกอบข้อความ

ต้องใช้พรสวรรค์ในการสร้างแอปพลิเคชันมือถือ แต่ต้องใช้สไตล์เพื่อทำให้แอปพลิเคชันยอดเยี่ยม ถ้าคุณเป็นนักออกแบบกราฟิก คุณรู้เรื่องนี้โดยสัญชาตญาณ ลึกลงไปในกระดูกของคุณ ถ้าคุณเป็นนักพัฒนาคุณอาจจะคร่ำครวญและกลอกตา ไม่ว่าในกรณีใดเข้าใจ-
พื้นฐานของการจัดสไตล์ส่วนประกอบ Reactive Native เป็นสิ่งสำคัญในการสร้างแอปพลิเคชันที่น่าสนใจที่ผู้อื่นต้องการใช้
คุณมีประสบการณ์กับ CSS มาบ้างแล้ว แม้ว่าจะไม่มีอะไรมากไปกว่าเห็นไวยากรณ์ คุณสามารถเข้าใจได้ง่ายว่ากฎ CSS ชอบสีพื้นหลังอย่างไร:
'สีแดง' มีขึ้นเพื่อทำ เมื่อคุณเริ่มอ่านบทนี้ อาจดูเหมือนสไตล์
ing คอมโพเนนต์ใน React Native นั้นง่ายพอๆ กับการใช้ชื่อ camelCase สำหรับกฎ CSS ตัวอย่างเช่น การตั้งค่าสีพื้นหลังบนส่วนประกอบ React Native นั้นใช้เกือบไวยากรณ์เดียวกันbackgroundColor: 'red' —แต่ขอเตือนไว้ก่อนว่านี่คือที่ที่ซิม-
อิลาริตี้สิ้นสุด

หน้า 4

พยายามอย่ายึดติดกับสิ่งที่คุณทำใน CSS โอบกอดวิธี React Native
และคุณจะพบว่าการเรียนรู้วิธีจัดรูปแบบองค์ประกอบเป็นประสบการณ์ที่น่าพึงพอใจมากกว่า
ence—แม้กระทั่งสำหรับนักพัฒนา
ส่วนแรกของบทนี้จะให้ภาพรวมของส่วนประกอบสำหรับการจัดแต่ง ดี
ตรวจสอบให้แน่ใจว่าคุณเข้าใจวิธีต่างๆ ในการใช้สไตล์กับส่วนประกอบและอภิปราย
วิธีจัดระเบียบสไตล์ในแอปพลิเคชัน การสร้างนิสัยองค์กรที่ดีในตอนนี้จะ
ทำให้จัดการสิ่งต่าง ๆ ได้ง่ายขึ้น และจะอำนวยความสะดวกในการใช้เทคนิคขั้นสูงขึ้น
ลงที่ถนน
เนื่องจาก React Native ถูกจัดรูปแบบโดยใช้ JavaScript เราจะพูดถึงวิธีเริ่มคิด
ของสไตล์เป็นโค้ดและวิธีใช้ประโยชน์จากคุณลักษณะ JavaScript เช่น ตัวแปรและ func-
ชั่น สองส่วนสุดท้ายสำรวจองค์ประกอบมุมมองและองค์ประกอบข้อความเกี่ยวกับสไตล์ ใน
บางกรณี เราจะใช้ตัวอย่างสั้นๆ ในการอธิบายหัวข้อ แต่ส่วนใหญ่เราจะเดิน
ผ่านการจัดแต่งทรงผมของจริง คุณจะนำสิ่งที่คุณเรียนรู้ไปใช้กับโครงสร้าง-
ของบัตรโปรไฟล์
สำหรับโค้ดตัวอย่างทั้งหมดในบทนี้ คุณสามารถเริ่มต้นด้วยแอปที่สร้างโดยค่าเริ่มต้นและ
แทนที่เนื้อหาของ App.js ด้วยรหัสจากรายการแต่ละรายการ ที่มาที่สมบูรณ์
สามารถดูไฟล์ได้ที่www.manning.com/books/react-native-in-actionและในหนังสือ
ที่เก็บ Git ที่ https://github.com/dabit3/react-native-in-action ภายใต้บทที่ 4

4.1 การใช้และการจัดรูปแบบใน React Native

React Native มาพร้อมกับส่วนประกอบในตัวมากมาย และชุมชนได้สร้างมากมาย
คุณสามารถรวมเข้ากับโครงการของคุณได้มากขึ้น ส่วนประกอบรองรับชุดสไตล์เฉพาะ
สไตล์เหล่านั้นอาจใช้หรือไม่ใช้ได้กับส่วนประกอบประเภทอื่น ตัวอย่างเช่น,
ข้อความองค์ประกอบสนับสนุนfontWeightคุณสมบัติ (fontWeightหมายถึง thick-
ของฟอนต์) แต่คอมโพเนนต์Viewไม่มี ในทางกลับกันองค์ประกอบมุมมอง
รองรับคุณสมบัติflex (flexหมายถึงเลย์เอาต์ของส่วนประกอบภายในมุมมอง) แต่
ข้อความองค์ประกอบไม่ได้
องค์ประกอบสำหรับการจัดสไตล์บางอย่างจะคล้ายกันระหว่างส่วนประกอบต่างๆ แต่ไม่เหมือนกัน สำหรับ

ตัวอย่างองค์ประกอบViewรองรับคุณสมบัติshadowColorในขณะที่Text
ส่วนประกอบรองรับคุณสมบัติtextShadowColor บางสไตล์ เช่นShadowProp-
TypesIOS ใช้กับแพลตฟอร์มเฉพาะ (ในกรณีนี้ กับ iOS)
การเรียนรู้สไตล์ต่างๆ และวิธีจัดการกับมันต้องใช้เวลา นั่นเป็นเหตุผลว่าทำไม
สิ่งสำคัญคือต้องเริ่มต้นด้วยพื้นฐาน เช่น วิธีการใช้และจัดระเบียบสไตล์ ส่วนนี้
จะเน้นสอนพื้นฐานการจัดแต่งทรงผมเหล่านั้น แล้วคุณจะมีพื้นฐานที่ดี
เพื่อเริ่มสำรวจสไตล์และสร้างตัวอย่างองค์ประกอบการ์ดโปรไฟล์
เคล็ดลับ สำหรับข้อมูลอ้างอิงที่ชัดเจนเกี่ยวกับวิธีทำให้แอปบนอุปกรณ์เคลื่อนที่ใช้งานได้ โปรดดูที่ Matt Lacey's
เรื่องการใช้งาน (Manning, 2018; www.manning.com/books/usability-matters)

4.1.1 การใช้สไตล์ในแอปพลิเคชัน

เพื่อแข่งขันในตลาด แอพพลิเคชั่นมือถือต้องมีสไตล์ คุณสามารถ
พัฒนาแอพที่ทำงานได้อย่างสมบูรณ์ แต่ถ้ามันดูแย่และไม่มีส่วนร่วม คนจะไม่
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

77
การใช้และการจัดระเบียบสไตล์ใน React Native
จะสนใจ คุณไม่จำเป็นต้องสร้างแอพที่ดูร้อนแรงที่สุดในโลก
แต่คุณต้องมุ่งมั่นที่จะสร้างผลิตภัณฑ์ที่สวยงาม เนียนกริบ
แอปมีอิทธิพลอย่างมากต่อการรับรู้ของผู้คนเกี่ยวกับคุณภาพของแอป
คุณสามารถใช้สไตล์กับองค์ประกอบใน React Native ได้หลายวิธี ในบท
1 และ 3 เราไปดูการจัดสไตล์อินไลน์ (แสดงในรายการถัดไป) และจัดสไตล์โดยใช้
StyleSheet (รายการ 4.2)

```
นำเข้า React { ส่วนประกอบ } จาก 'react'
นำเข้า { ข้อความ, ดู } จาก 'react-native'
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์ {
แสดงผล () {
กลับ (
<ดูสไตล์={{marginLeft: 20, marginTop: 20}}>
<Text style={{fontSize: 18,color: 'red'}}>ข้อความบางส่วน</Text>
</ดู>
)
}
}
```

อย่างที่คุณเห็น คุณสามารถระบุสไตล์ได้หลายแบบพร้อมกันโดยระบุวัตถุให้
รูปแบบสถานที่ให้บริการ

```
นำเข้า React { ส่วนประกอบ } จาก 'react'
นำเข้า { StyleSheet, Text, View } จาก 'react-native'
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์ {
แสดงผล () {
กลับ (
<ดู style={styles.container}>
<Text style={[styles.message,styles.warning]}>ข้อความบางส่วน</Text>
</ดู>
)
}
}
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ระยะขอบซ้าย: 20,
ขอบด้านบน: 20
},
ข้อความ: {
ขนาดตัวอักษร: 18
},
คำเตือน: {
สี: 'แดง'
}
});
```

ใช้รูปแบบอินไลน์กับ a
ตอบสนององค์ประกอบดั้งเดิม
ใช้รูปแบบอินไลน์หลายแบบพร้อมกัน
อ้างอิงคอนเทนเนอร์ style
กำหนดไว้ในสไตล์ชีตสไตล์
ใช้อาร์เรย์เพื่ออ้างอิง
ทั้งข้อความและคำเตือน
สไตล์จากสไตล์ชีต
กำหนดสไตล์โดยใช้ StyleSheet.create
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

ตามหน้าที่ ไม่มีความแตกต่างระหว่างการใช้รูปแบบอินไลน์กับการอ้างอิง a
รูปแบบที่กำหนดไว้ในStyleSheet ด้วยStyleSheetคุณสร้างวัตถุสไตล์และอ้างอิงถึง
แต่ละสไตล์เป็นรายบุคคล แยกรูปแบบจากการแสดงผลวิธีการทำให้โค้ด
เข้าใจได้ง่ายขึ้นและส่งเสริมการนำสไตล์มาใช้ซ้ำในทุกองค์ประกอบ
เมื่อใช้ชื่อรูปแบบ เช่นคำเตือนจะเป็นเรื่องง่ายที่จะระบุเจตนาของข้อความ
แต่สีสไตล์อินไลน์: 'สีแดง'ไม่ได้ให้ข้อมูลเชิงลึกว่าทำไมข้อความจึงเป็นสีแดง มี
สไตล์ที่ระบุในที่เดียวแทนที่จะเป็นแบบอินไลน์ในส่วนประกอบจำนวนมากทำให้ง่ายต่อการ
ใช้การเปลี่ยนแปลงทั่วทั้งแอปพลิเคชัน ลองนึกภาพคุณต้องการเปลี่ยนข้อความเตือน-
ปราชญ์เป็นสีเหลือง สิ่งที่คุณต้องทำคือเปลี่ยนการกำหนดสไตล์ครั้งเดียวในสไตล์ชีต
สี: 'เหลือง' .
รายการ 4.2 ยังแสดงวิธีการระบุหลายสไตล์โดยระบุอาร์เรย์ของสไตล์
คุณสมบัติ. จำไว้ว่าเมื่อทำเช่นนี้ สไตล์ล่าสุดที่ส่งเข้ามาจะแทนที่
สไตล์ก่อนหน้าหากมีคุณสมบัติที่ซ้ำกัน ตัวอย่างเช่น ถ้าอาร์เรย์ของสไตล์แบบนี้คือ
ที่ให้มา ค่าสุดท้ายสำหรับสีจะแทนที่ค่าก่อนหน้าทั้งหมด:

```
style={[{color: 'black'},{color: 'yellow'},{สี: 'สีแดง'}]}
```

ในตัวอย่างนี้ สีจะเป็นสีแดง
นอกจากนี้ยังสามารถรวมทั้งสองวิธีเข้าด้วยกันได้โดยการระบุอาร์เรย์ของการจัดสไตล์
คุณสมบัติโดยใช้สไตล์อินไลน์และการอ้างอิงไปยังสไตล์ชีต:

```
style={[{สี: 'black'}, styles.message]}
```

React Native นั้นมีความยืดหยุ่นสูงในเรื่องนี้ ซึ่งสามารถเป็นได้ทั้งดีและไม่ดี ระบุ-
การใช้สไตล์อินไลน์เมื่อคุณพยายามสร้างต้นแบบบางอย่างอย่างรวดเร็วนั้นง่ายมาก
แต่ในระยะยาว คุณจะต้องระมัดระวังในการจัดระเบียบสไตล์ของคุณ มิฉะนั้น
ใบสมัครของคุณอาจกลายเป็นเรื่องยุ่งเหยิงและจัดการได้ยาก โดยการจัดระเบียบของคุณ
สไตล์ คุณจะทำสิ่งต่อไปนี้ได้ง่ายขึ้น:

- รักษา codebase ของแอปพลิเคชันของคุณ
- ใช้สไตล์ซ้ำระหว่างส่วนประกอบต่างๆ
- ทดลองเปลี่ยนสไตล์ระหว่างการพัฒนา

4.1.2 รูปแบบการจัดวาง

อย่างที่คุณอาจสงสัยจากส่วนก่อนหน้านี้ การใช้สไตล์อินไลน์ไม่ใช่สิ่งแนะนำ
แนวทางแก้ไข: สไตล์ชีตเป็นวิธีที่มีประสิทธิภาพมากกว่าในการจัดการสไตล์ แต่
ในทางปฏิบัติหมายความว่าอย่างไร
เมื่อจัดรูปแบบเว็บไซต์ เราใช้สไตล์ชีตตลอดเวลา บ่อยครั้งที่เราใช้เครื่องมือเช่น Sass

Less และ PostCSS เพื่อสร้างสไตล์ชีตแบบเสาหินสำหรับแอปพลิเคชันทั้งหมด ใน
โลกของเว็บ สไตล์มีความสำคัญทั่วโลก แต่นั่นไม่ใช่วิธี React Native
React Native มุ่งเน้นไปที่องค์ประกอบ เป้าหมายคือการทำให้ส่วนประกอบต่างๆ สามารถนำกลับมาใช้ใหม่ได้
และสแตนด์อโลนให้ได้มากที่สุด มีส่วนประกอบขึ้นอยู่กับสไตล์ชีตของแอปพลิเคชัน
เป็นสิ่งที่ตรงกันข้ามของโมดูลาร์ ใน React Native สไตล์ถูกกำหนดขอบเขตไว้ที่ส่วนประกอบ—ไม่ใช่ to
แอปพลิเคชัน
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 7

79
การใช้และการจัดระเบียบสไตล์ใน React Native
วิธีที่จะทำให้การห่อหุ้มนี้สำเร็จขึ้นอยู่กับความชอบของทีมคุณ
ไม่มีทางถูกหรือผิด แต่ในชุมชน React Native คุณจะพบกับสิ่งที่เหมือนกันสองอย่าง
วิธีการ:

- ประกาศสไตล์ชีตในไฟล์เดียวกับส่วนประกอบ
- ประกาศสไตล์ชีตในไฟล์แยกต่างหาก ภายนอกคอมโพเนนต์

D ประกาศสไตล์ชีตในไฟล์เดียวกับส่วนประกอบ
ดังที่คุณได้ทำในหนังสือเล่มนี้แล้ว วิธีที่นิยมในการประกาศสไตล์อยู่ในองค์ประกอบ
เน็ตที่จะใช้พวกเขา ประโยชน์หลักของวิธีนี้คือส่วนประกอบ
และรูปแบบของมันถูกห่อหุ้มไว้อย่างสมบูรณ์ในไฟล์เดียว ส่วนประกอบนี้สามารถแล้ว
ย้ายหรือใช้ที่ใดก็ได้ในแอพ นี่เป็นแนวทางทั่วไปสำหรับส่วนประกอบ
การออกแบบที่คุณจะได้เห็นบ่อยๆในชุมชน React Native
เมื่อรวมคำจำกัดความของสไตล์ชีต
ด้วยองค์ประกอบตามแบบแผนทั่วไป
คือการกำหนดลักษณะหลังส่วนประกอบ
รายการทั้งหมดในหนังสือเล่มนี้มีจนถึงตอนนี้
ลดการประชุมนี้
D ECLARING สไตล์ชีตในไฟล์แยกต่างหาก
หากคุณเคยเขียน CSS ให้ใส่ your
ลักษณะเป็นไฟล์แยกต่างหากอาจดูเหมือน a
แนวทางที่ดีขึ้นและรู้สึกคุ้นเคยมากขึ้น NS
คำจำกัดความของสไตล์ชีตถูกสร้างขึ้นใน
ไฟล์อัตรา ตั้งชื่ออะไรก็ได้ตามใจชอบ

(styles.js เป็นเรื่องปกติ) แต่ต้องแน่ใจว่านามสกุล
คือ .js; มันคือ JavaScript หลังจากทั้งหมด สไตล์ชีต
ไฟล์และไฟล์ส่วนประกอบจะถูกบันทึกใน
โฟลเดอร์เดียวกัน
โครงสร้างไฟล์ตามภาพ
4.1 รักษาความสัมพันธ์ที่ใกล้ชิดระหว่าง
ส่วนประกอบและรูปแบบและกำบังเล็กน้อย
ความชัดเจนโดยไม่ผสมคำจำกัดความของสไตล์กับ
ลักษณะการทำงานของส่วนประกอบ
รายการ 4.3 สอดคล้องกับไฟล์ styles.js ที่
จะถูกนำมาใช้เพื่อจัดรูปแบบองค์ประกอบเช่น
ComponentA และ ComponentB ในรูป
ใช้ชื่อที่มีความหมายเมื่อกำหนดของคุณ
สไตล์ชีต ดังนั้นจึงชัดเจนว่าส่วนใดขององค์ประกอบ
เน็ตกำลังถูกจัดสไตล์
MYREACTPROJECT
ส่วนประกอบ
ส่วนประกอบ
COMPONENTA.JS
APP.JS
INDEX.JS
STYLES.JS
ส่วนประกอบB
COMPONENTB.JS
STYLES.JS
รูปที่ 4.1 ตัวอย่างโครงสร้างไฟล์ด้วย
รูปแบบที่แยกจากส่วนประกอบในเครื่องเดียว
โฟลเดอร์แทนที่จะเป็นไฟล์เดียว
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 8

นำเข้า { StyleSheet } จาก 'react-native'
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ขอบบน: 150,

```
backgroundColor: '#ededed',
flexWrap: 'ห่อ'
}
})
ปุ่ม const = StyleSheet.create ({
หลัก: {
ดิ้น: 1,
ความสูง: 70,
พื้นหลังสี: 'สีแดง',
justifyContent: 'ศูนย์',
alignItems: 'ศูนย์',
ระยะขอบซ้าย: 20,
ขอบขวา: 20
}
})
ส่งออก { สไตล์ ปุ่ม }
```

คอมโพเนนต์จะนำเข้าสไตล์ชีตภายนอกและสามารถอ้างอิงสไตล์ที่กำหนดไว้ได้
ภายในพวกเขา

```
นำเข้า { สไตล์, ปุ่ม } จาก './component/styles'
<ดู style={styles.container}>
<TouchableHighlight style={buttons.primary} />
...
</TouchableHighlight>
</ดู>
```

4.1.3 ลักษณะเป็นรหัส

คุณได้เห็นแล้วว่า JavaScript ใช้เพื่อกำหนดสไตล์ใน React Native อย่างไร แม้จะมีค-
ด้วยภาษาสคริปต์เต็มรูปแบบพร้อมตัวแปรและฟังก์ชัน สไตล์ของคุณค่อนข้างจะเป็นแบบนั้น
คงที่ แต่พวกเขาไม่จำเป็นต้องเป็นอย่างแน่นอน!
นักพัฒนาเว็บได้ต่อสู้กับ CSS มาหลายปีแล้ว เทคโนโลยีใหม่ๆ เช่น Sass, Less และ
PostCSS ถูกสร้างขึ้นเพื่อแก้ไขข้อจำกัดมากมายของสไตล์ชีตแบบเรียงซ้อน
แม้แต่เรื่องง่ายๆ เช่น การกำหนดตัวแปรเพื่อเก็บสีหลักของเว็บไซต์ก็คือ
เป็นไปไม่ได้หากไม่มีตัวประมวลผลล่วงหน้า CSS คุณสมบัติที่กำหนดเอง CSS สำหรับ Cascading Vari-
สามารถแนะนำผู้สมัครระดับโมดูล 1 ในเดือนธันวาคม 2015 ได้แนะนำ
แนวคิดของคุณสมบัติที่กำหนดเองซึ่งคล้ายกับตัวแปร แต่ในขณะที่เขียน
มีการใช้งานเบราว์เซอร์น้อยกว่า 80% รองรับฟังก์ชันนี้
สร้างสไตล์ชีตและบันทึก
มันอยู่ในรูปแบบคงที่
กำหนดสไตล์สำหรับคอนเทนเนอร์

สามารถอ้างอิงได้โดย
องค์ประกอบเป็น styles.container
สร้างสไตล์ชีตที่สอง และ
บันทึกไว้ในปุ่มคงที่
กำหนดสไตล์สำหรับปุ่มหลัก
มันสามารถอ้างอิงโดยส่วนประกอบ
เป็นปุ่มหลัก
ส่งออกทั้งสไตล์และปุ่ม
สไตล์ชีตเพื่อให้องค์ประกอบมี
เข้าถึงค่าคงที่
นำเข้าหลายสไตล์ชีต
ส่งออกจาก styles.js
อ้างอิง
กับสไตล์
คอนเทนเนอร์
สไตล์ที่สร้างขึ้น
ในสไตล์.js
อ้างอิงถึงปุ่มหลัก
สไตล์ที่สร้างใน styles.js
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 9

81
การใช้และการจัดระเบียบสไตล์ใน React Native
มาใช้ประโยชน์จากความจริงที่ว่าเรากำลังใช้ JavaScript และเริ่มคิดถึงสไตล์
เป็นรหัส คุณจะสร้างแอปพลิเคชันง่าย ๆ ที่ให้ผู้ใช้ปุ่มเพื่อเปลี่ยน
ธีมจากสว่างไปมืด แต่ก่อนที่คุณจะเริ่มเขียนโค้ด มาดูสิ่งที่คุณเป็น
พยายามที่จะสร้าง
แอปพลิเคชั่นมีปุ่มเดียวบนหน้าจอ ปุ่มนั้นถูกล้อมรอบด้วยเล็ก
กล่องสี่เหลี่ยม เมื่อกดปุ่ม ธีมจะสลับไปมา เมื่อธีมแสง
เลือกแล้ว ป้ายปุ่มจะขึ้นว่า สีขาว พื้นหลังจะเป็นสีขาว และกล่อง
รอบปุ่มจะเป็นสีดำ เมื่อเลือกธีมสีเข้ม ป้ายปุ่มจะ

พูดว่า Black พื้นหลังจะเป็นสีดำ และกล่องรอบๆ ปุ่มจะเป็นสีขาว
รูปที่ 4.2 แสดงว่าหน้าจอควรเป็นอย่างไรเมื่อเลือกธีม
สำหรับตัวอย่างนี้ จัดระเบียบสไตล์ในไฟล์ที่แยกจากกัน คือ styles.js จากนั้นสร้างคอน
กำหนดให้เก็บค่าสีไว้ และสร้างสไตล์ชีตสองชุดสำหรับธีมสีอ่อนและสีเข้ม

นำเข้า {StyleSheet} จาก 'react-native';
ส่งออกสี const = {
สีดำ',
แสง: 'สีขาว'
};
const baseContainerStyles = {
ดิ้น: 1,
justifyContent: 'ศูนย์',
alignItems: 'ศูนย์'
};
const baseBoxStyles = {
justifyContent: 'ศูนย์',
alignItems: 'ศูนย์',
ขอบกว้าง: 2,
ความสูง: 150,
ความกว้าง: 150
};
const lightStyleSheet = StyleSheet.create ({

คลิกขาว
ปุ่ม
คลิกสีดำ
ปุ่ม
รูปที่ 4.2 แอปพลิเคชั่นอย่างง่ายที่รองรับสองธีม
ขาวกับดำ. ผู้ใช้สามารถกดปุ่มเพื่อสลับระหว่าง
พื้นหลังสีขาวและพื้นหลังสีดำ
กำหนดสีที่จะคงที่อย่างต่อเนื่อง
สอดคล้องกับธีมแสงและความมืด
วัตถุ JavaScript ที่จะถือ
รูปแบบคอนเทนเนอร์ฐาน
วัตถุ JavaScript ที่จะถือ
รูปแบบกล่องฐาน
สร้างสไตล์ชีต
สำหรับธีมไฟ

หน้า 10


```
คอนเทนเนอร์: {
...baseContainerStyles,
backgroundColor: Colors.light
},
กล่อง: {
...baseBoxStyles,
borderColor: Colors.dark
}
});
const darkStyleSheet = StyleSheet.create ({
คอนเทนเนอร์: {
...baseContainerStyles,
backgroundColor: Colors.dark
},
กล่อง: {
...baseBoxStyles,
borderColor: Colors.light
}
});
ส่งออกฟังก์ชันเริ่มต้น getStyleSheet (useDarkTheme) {
ส่งคืน useDarkTheme ? darkStyleSheet : lightStyleSheet;
}
```

เมื่อกำหนดค่าสไตล์แล้ว คุณสามารถเริ่มสร้างแอปส่วนประกอบในแอพได้
เจเอส เพราะคุณมีเพียงรูปแบบแสงและสีสร้างฟังก์ชั่นยูทิลิตี้getStyleSheet ,
ซึ่งใช้ค่าบูลีน หากระบุเป็น trueธีมสีเข้มจะถูกส่งคืน อื่น ๆ-
ฉลาดธีมแสงจะถูกส่งกลับ

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { ปุ่ม, สไตล์ชีต, ดู } จาก 'react-native';
นำเข้า getStyleSheet จาก './styles';
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์ {
ตัวสร้าง (อุปกรณ์ประกอบฉาก) {
สุดยอด(อุปกรณ์ประกอบฉาก);
this.state = {
darkTheme: เท็จ
};
this.toggleTheme = this.toggleTheme.bind (นี่);
}
```

สลับธีม () {
this.setState({darkTheme: !this.state.darkTheme})
};
เรนเดอร์ () {
รูปแบบ const = getStyleSheet (this.state.darkTheme);
const backgroundColor =
สร้างสไตล์ชีต
สำหรับธีมมืด
ฟังก์ชั่นที่
จะกลับมา
เหมาะสม
ตามธีม
บนบูลีน
ค่า
คืนความมืด
ธีมถ้าใช้DarkTheme
เป็นความจริง; มิฉะนั้น
ส่งคืนธีมแสง
นำเข้าฟังก์ชัน getStyleSheet
จากรูปแบบภายนอก
เริ่มต้นสถานะของส่วนประกอบ
เพื่อแสดงชุดรูปแบบแสงโดยค่าเริ่มต้น
เพื่อหลีกเลี่ยงข้อยกเว้น
ฟังก์ชัน toggleTheme ต้อง
ถูกผูกไว้กับองค์ประกอบ
สลับ
ค่าธีม
ในสถานะ
เมื่อใดก็ตามที่
ฟังก์ชันคือ
เรียกว่า
ใช้ฟังก์ชัน getStyleSheet ที่นำเข้า
เพื่อรับสไตล์ชีตที่เหมาะสมสำหรับ
ธีมไหนก็ควรแสดง
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

83

ส่วนประกอบมุมมองจัดแต่งทรงผม

StyleSheet.flatten(styles.container).พื้นหลังสี;

กลับ (

<ดู style={styles.container}>

<ดู style={styles.box}>

<ชื่อปุ่ม={backgroundColor}

onPress={this.toggleTheme}/>

</ดู>

</ดู>

);

}

}

แอปพลิเคชันสลับธีม: อย่าลังเลที่จะทดลองและก้าวต่อไปอีกเล็กน้อย ลอง
เปลี่ยนธีมแสงเป็นสีอื่น สังเกตง่ายแค่ไหน เพราะสี
ถูกกำหนดให้เป็นค่าคงที่ในที่เดียว ลองเปลี่ยนป้ายปุ่มในธีมสีเข้ม
ให้เป็นสีเดียวกับพื้นหลังแทนที่จะเป็นสีขาวเสมอ ลองสร้างทั้งหมด
ธีมใหม่ หรือแก้ไขโค้ดเพื่อรองรับธีมต่างๆ มากมาย แทนที่จะเป็นเพียงสองธีม—
มีความสุข!

4.2 ส่วนประกอบการจัดแต่งทรงผม

ตอนนี้คุณมีภาพรวมที่เหมาะสมของการจัดสไตล์ใน React Native แล้ว มาพูดถึงกันมากขึ้น
สไตล์ส่วนบุคคล บทนี้ครอบคลุมคุณสมบัติพื้นฐานมากมายที่คุณจะใช้
เวลา. ในบทที่ 5 เราจะเจาะลึกและแนะนำสไตล์ที่คุณจะไม่เคยเห็น
วันและสไตล์ที่เจาะจงแพลตฟอร์ม แต่สำหรับตอนนี้ มาเน้นที่พื้นฐาน: ในนี้
ส่วนที่เป็นส่วนประกอบดู ดูองค์ประกอบเป็นกลุ่มอาคารหลัก
ของ UI และเป็นหนึ่งในองค์ประกอบที่สำคัญที่สุดที่ต้องทำความเข้าใจเพื่อให้ได้สไตล์ของคุณ
ขวา. โปรดจำไว้ว่าองค์ประกอบViewนั้นคล้ายกับแท็กdiv ของ HTML ในแง่ที่คุณ
สามารถใช้เพื่อห่อองค์ประกอบอื่น ๆ และสร้างบล็อกของรหัส UI ในนั้น
ในขณะที่คุณดำเนินไปตามบท คุณจะใช้สิ่งที่คุณได้เรียนรู้เพื่อสร้างความจริง
ส่วนประกอบ: การ์ดโปรไฟล์ การสร้างโปรไฟล์การ์ดจะแสดงวิธีการใส่ทุกอย่าง
ด้วยกัน. รูปที่ 4.3 แสดงว่าส่วนประกอบจะมีลักษณะอย่างไรเมื่อสิ้นสุดส่วนนี้
ในกระบวนการสร้างองค์ประกอบนี้ คุณจะได้เรียนรู้วิธีดำเนินการดังต่อไปนี้:

- สร้างเส้นขอบรอบคอนเทนเนอร์โปรไฟล์โดยใช้borderWidth
- ปัดเศษมุมของเส้นขอบนั้นด้วยborderRadius

▪ สร้างเส้นขอบที่ดูเหมือนวงกลมโดยใช้borderRadius ขนาดครึ่งหนึ่งของ
ความกว้างของส่วนประกอบ
▪ วางตำแหน่งทุกอย่างโดยใช้คุณสมบัติระยะขอบและช่องว่างภายใน
ในส่วนต่อไปจะสอนเทคนิคการจัดแต่งทรงผมที่คุณจำเป็นต้องรู้เพื่อสร้าง
ส่วนประกอบการ์ดโปรไฟล์ เราจะเริ่มง่ายๆ โดยพูดถึงวิธีตั้งค่าส่วนประกอบ
สีพื้นหลัง. คุณจะสามารถใช้เทคนิคเดียวกันนั้นในการตั้งค่าพื้นหลังได้
สีของบัตรโปรไฟล์
ยูทิลิตี้ React Native StyleSheet.flatten
ฟังก์ชั่นแปลงวัตถุ StyleSheet
ลงในวัตถุ JavaScript ซึ่งทำให้It
ง่ายกว่ามากในการรับ backgroundColor
อ้างอิง
ธีมของ
คอนเทนเนอร์
สไตล์
อ้างอิง
รูปแบบกล่องของธีม
(ขอบกล่อง
รอบปุ่ม)
การแสดงสตริง
ของความเป็นสี
ใช้โดยธีม
เมื่อกดปุ่มโทรออก
ฟังก์ชัน toggleTheme เป็น
สลับจากธีมหนึ่งไปอีกธีมหนึ่ง
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 12

4.2.1 การตั้งค่าสีพื้นหลัง

หากไม่มีสีสัน อินเทอร์เฟซผู้ใช้ (UI) จะดูน่าเบื่อและน่าเบื่อ คุณไม่จำเป็นต้อง

การระเบิดของสีเพื่อทำให้สิ่งต่าง ๆ ดูน่าสนใจ แต่คุณต้องการเล็กน้อย NS
คุณสมบัติ backgroundColor กำหนดสีพื้นหลังขององค์ประกอบ คุณสมบัตินี้
รับสตริงของคุณสมบัติอย่างใดอย่างหนึ่งที่แสดงในตาราง 4.1 สีเดียวกันสามารถใช้ได้ -
สามารถแสดงข้อความบนหน้าจอได้เช่นกัน
ตาราง 4.1 รูปแบบสีที่รองรับ

#rgb
'#06f'
#rgba
'#06เอฟซี'
#rrggbb
'#0066ff'
#rrggbbaa
'#ff00ff00'
rgb(หมายเลข หมายเลข หมายเลข)
'rgb(0, 102, 255)'
rgb(ตัวเลข, ตัวเลข, ตัวเลข, อัลฟ่า)
'rgba(0, 102, 255, .5)'
hsl(สี ความอิ่มตัว ความสว่าง)
'hsl(216, 100%, 50%)'
hsla(ฮิว, ความอิ่มตัว, ความสว่าง, อัลฟ่า)
'hsla(216, 100%, 50%, .5)'
พื้นหลังโปร่งใส
'โปร่งใส'
สีที่ระบุชื่อ CSS3 ใดๆ (ดำ แดง น้ำเงิน และอื่นๆ)
'ดอดเจอร์บลู'
สีพื้นหลัง

มุมโค้งมน

borderWidth

ศูนย์กลาง

แนวนอน

borderRadius half

ความกว้าง

รูปที่ 4.3 ส่วนประกอบ Profile Card หลังโครงสร้าง
ดูส่วนประกอบที่ได้รับการจัดรูปแบบ บัตรโปรไฟล์คือ a
สี่เหลี่ยมผืนผ้าที่มีมุมมนและส่วนวงกลมสำหรับ a
รูปโปรไฟล์

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 13

85

ส่วนประกอบมุมมองจัดแต่งทรงผม

โชคดีที่รูปแบบสีที่รองรับเป็นสีเดียวกับที่ CSS รองรับ เรา
จะไม่ลงรายละเอียดมาก แต่เพราะนี่อาจเป็นครั้งแรกที่คุณได้เห็น
รูปแบบเหล่านี้ มีคำอธิบายโดยย่อ:

- rgb ย่อมาจากสีแดง สีเขียว และสีน้ำเงิน คุณสามารถระบุค่าสีแดง สีเขียว และ
สีน้ำเงินโดยใช้มาตราส่วนตั้งแต่ 0–255 (หรือเลขฐานสิบหก 00–ff) ตัวเลขที่สูงขึ้นหมายถึง
แต่ละสีมากขึ้น
- อัลฟาคล้ายกับความทึบ (0 คือโปร่งใส 1 คือทึบ)
- สีแทน 1 องศาบนวงล้อสี 360 องศา โดยที่ 0 คือสีแดง 120 คือสีเขียว
และ 240 เป็นสีน้ำเงิน
- ความอิ่มตัวคือความเข้มของสีจากเฉดสีเทา 0% ไปจนถึงสีเต็ม 100%
- ความสว่างเป็นเปอร์เซ็นต์ระหว่าง 0% ถึง 100% 0% เข้มขึ้น (ใกล้กับสีดำมากขึ้น)
และแสง 100% (ใกล้สีขาวมากขึ้น)

คุณเคยเห็นbackgroundColorถูกนำไปใช้ในตัวอย่างก่อนหน้านี้ ดังนั้น มาเริ่มขั้นตอนกัน
ในตัวอย่างต่อไป เพื่อใช้ทักษะใหม่ของคุณเพื่อสร้างสิ่งที่เป็นจริง มาเริ่มสร้าง-
ในบัตรโปรไฟล์ ตอนนี้จะดูไม่มากอย่างที่คุณเห็นในภาพที่ 4.4—คือ
แค่สี่เหลี่ยมสีขนาด 300 × 400
รายการต่อไปนี้แสดงรหัสเริ่มต้น ไม่ต้องกังวลกับความจริงที่ว่าส่วนใหญ่
ไม่มีส่วนเกี่ยวข้องกับการจัดแต่งทรงผม เราจะเดินผ่านแต่ละชิ้น แต่คุณต้องมีพื้นฐาน-
ที่จะเริ่มต้น

รูปที่ 4.4 สี่เหลี่ยมสีธรรมดา 300 × 400
ที่เป็นฐานของส่วนประกอบการ์ดโปรไฟล์

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 14

86

C HAPTER 4 รู้เบื้องต้นเกี่ยวกับสไตล์

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { StyleSheet, View} จาก 'react-native';
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
```

```
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<ดู style={styles.cardContainer}/>
</ดู>
);
}
}
const profileCardColor = 'dodgerblue';
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ดิ้น: 1,
justifyContent: 'ศูนย์',
alignItems: 'ศูนย์'
},
คอนเทนเนอร์การ์ด: {
พื้นหลังสี: profileCardColor,
ความกว้าง: 300,
ส่วนสูง: 400
}
});
```

องค์ประกอบViewแรกเป็นองค์ประกอบที่อยู่นอกสุด ทำหน้าที่เป็นภาชนะรอบๆ
อย่างอื่น. จุดประสงค์เพียงอย่างเดียวคือการจัดวางส่วนประกอบย่อยไว้บนจอแสดงผลของอุปกรณ์
องค์ประกอบมุมมองที่สองจะเป็นคอนเทนเนอร์สำหรับการ์ดโปรไฟล์ สำหรับตอนนี้มันคือ
สี่เหลี่ยมสี 300 × 400

4.2.2 การตั้งค่าคุณสมบัติเส้นขอบ

การใช้สีพื้นหลังกับส่วนประกอบทำให้ดูโดดเด่นอย่างแน่นอน แต่ด้วย-
ออกเส้นขอบที่คมชัดซึ่งวาดขอบของส่วนประกอบ ดูเหมือนว่าคอม
ponent ลอยอยู่ในอวกาศ การแบ่งองค์ประกอบที่ชัดเจนจะช่วยผู้ใช้
เข้าใจวิธีโต้ตอบกับแอปพลิเคชันมือถือของคุณ
การเพิ่มเส้นขอบรอบ ๆ ส่วนประกอบเป็นวิธีที่ดีที่สุดในการทำให้องค์ประกอบของหน้าจอ
ครีตความรู้สึกที่แท้จริง มีคุณสมบัติเส้นขอบค่อนข้างน้อยแต่ตามแนวคิดแล้วมี
เพียงสี่: borderColor , borderRadius , BorderStyleและborderWidth ที่เหมาะสมเหล่านี้-
ความสัมพันธ์นำไปใช้กับองค์ประกอบโดยรวม
สำหรับสีและความกว้าง จะมีคุณสมบัติเฉพาะสำหรับแต่ละด้านของเส้นขอบ:
borderTopColor , borderRightColor , borderBottomColor , borderLeftColor ,บอร์
derTopWidth , borderRightWidth , borderBottomWidth และborderLeftWidth สำหรับ
รัศมีขอบมีคุณสมบัติสำหรับแต่ละมุม: borderTopRightRadius , border-
BottomRightRadius , borderBottomLeftRadius และborderTopLeftRadius แต่มี

เพียงหนึ่งเส้นขอบ
องค์ประกอบมุมมองด้านนอกสุด
อ้างอิงถึงรูปแบบคอนเทนเนอร์ที่
จัดองค์ประกอบลูกดูให้อยู่ตรงกลาง
องค์ประกอบมุมมองภายใน
จะกลายเป็นโปรไฟล์
ส่วนประกอบของการ์ด
กำหนดสีสำหรับโปรไฟล์
การ์ดในตัวแปรในกรณีที่คุณต้องการ
เพื่อใช้ในที่ต่างๆ มากกว่าหนึ่งแห่ง
คำจำกัดความของสไตล์สำหรับ
ภาชนะชั้นนอกสุด
คำจำกัดความของสไตล์สำหรับ
คอนเทนเนอร์การ์ดโปรไฟล์
ตั้งค่าโปรไฟล์การ์ด
พื้นหลังสีเพื่อ
ค่าคงที่ก่อนหน้า
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 15

87
ส่วนประกอบมุมมองจัดแต่งทรงผม
C REATING ขอบด้วยสี , ความกว้างและรูปแบบคุณสมบัติ
ในการตั้งค่าเส้นขอบคุณต้องตั้งค่าborderWidthก่อน borderWidthคือขนาดของเส้นขอบ
และเป็นตัวเลขเสมอ คุณสามารถตั้งค่าborderWidthที่ใช้กับทั้งหมดได้
คอมโพเนนท์หรือเลือกขอบเขตความกว้างที่คุณต้องการกำหนดโดยเฉพาะ (บน ขวา บอท-
ทอมหรือซ้าย) คุณสามารถรวมคุณสมบัติเหล่านี้ได้หลายวิธีเพื่อให้ได้เอฟเฟกต์
คุณชอบ. ดูรูปที่ 4.5 สำหรับตัวอย่างบางส่วน
อย่างที่คุณเห็น คุณสามารถรวมสไตล์เส้นขอบเพื่อสร้างเอฟเฟกต์เส้นขอบรวมกันได้
รายการถัดไปแสดงให้เห็นว่าสิ่งนี้ทำได้ง่ายเพียงใด

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { StyleSheet, Text, View} จาก 'react-native';
```

```
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<รูปแบบตัวอย่าง={{borderWidth: 1}}>
<Text>ความกว้างขอบ: 1</Text>
</ตัวอย่าง>
<รูปแบบตัวอย่าง={{borderWidth: 3, borderLeftWidth: 0}}>
<ข้อความ>ความกว้างขอบ: 3, ความกว้างขอบด้านซ้าย: 0</Text>
</ตัวอย่าง>
<รูปแบบตัวอย่าง={{borderWidth: 3, borderLeftColor: 'red'}}>
<Text>ความกว้างขอบ: 3, borderLeftColor: 'สีแดง'</Text>
</ตัวอย่าง>
<รูปแบบตัวอย่าง={{borderLeftWidth: 3}}>
<Text>borderLeftWidth: 3</Text>
</ตัวอย่าง>
<example style={{borderWidth: 1, borderStyle: 'dash'}}>
<Text>ความกว้างขอบ: 1, สไตล์เส้นขอบ: 'ประ'</Text>
</ตัวอย่าง>
</ดู>
);
}
}
const ตัวอย่าง = (อุปกรณ์ประกอบฉาก) => (
<ดู style={[styles.example,props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</ดู>
);
```

รูปที่ 4.5 ตัวอย่างการรวมเส้นขอบต่างๆ

การตั้งค่าสไตล์

ตั้งค่า borderWidth เป็น 1

เพิ่ม borderWidth เป็น 3

ลบขอบด้านซ้ายและ

ตั้งค่า borderLeftWidth เป็น 0

ตั้งค่า borderWidth

ถึง 3 เพิ่มกลับ

ขอบซ้าย และ set

สีเป็นสีแดง

ตั้งค่าเฉพาะเส้นขอบด้านซ้ายด้วย

borderLeftWidth ตั้งค่าเป็น 3

เปลี่ยน borderStyle จาก

ค่าทึบเริ่มต้นเป็นเส้นประ
ส่วนประกอบตัวอย่างที่นำกลับมาใช้ใหม่ได้ด้วย a
ชุดสไตล์เริ่มต้นที่สามารถเป็นได้อย่างง่ายดาย
แทนที่ด้วยการส่งคุณสมบัติสไตล์
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 16

รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ดิ้น: 1,
justifyContent: 'ศูนย์',
alignItems: 'ศูนย์'
},
ตัวอย่าง: {
ขอบด้านล่าง: 15
}
});
เมื่อระบุเฉพาะborderWidthค่าเริ่มต้นborderColorเป็น'black'และborderStyle
เริ่มต้นที่'แข็ง' หากborderWidthหรือborderColorถูกตั้งค่าไว้ที่ระดับส่วนประกอบ
คุณสมบัติสามารถแทนที่ได้โดยใช้คุณสมบัติเฉพาะมากขึ้นเช่นborderWidthLeft ;
ความจำเพาะมีความสำคัญเหนือกว่าลักษณะทั่วไป
หมายเหตุborderStyle นั้นค่อนข้างบั๊กและฉันขอแนะนำให้ใช้ค่าเริ่มต้นที่เป็นของแข็ง
ชายแดน. หากคุณพยายามเปลี่ยนความกว้างเส้นขอบของด้านใดด้านหนึ่งและมีborderStyle
ตั้งค่าเป็น'จุด'หรือ'ประ'คุณจะได้รับข้อผิดพลาด นี้อาจจะได้รับการแก้ไขที่
ถึงจุดหนึ่ง แต่สำหรับตอนนี้อย่าใช้เวลามากเกินไปในการเกาหัวถ้า
borderStyle ไม่ทำงานตามที่คุณคาดหวัง ไฟล์ที่ออกไปในสมองของคุณและ
ไปกันเถอะ

U SING BORDER RADIUS เพื่อสร้างรูปร่าง

คุณสมบัติชายแดนอื่นที่สามารถนำมาใช้เพื่อผลที่ดีเป็นborderRadius มากมาย
วัตถุในโลกแห่งความเป็นจริงมีขอบตรง แต่ไม่ค่อยมีเส้นตรงสื่อถึงอะไร
ความรู้สึกของสไตล์ คุณจะไม่ซื้อรถยนต์ที่ดูเหมือนกล่อง คุณต้องการของคุณ
รถให้มีเส้นโค้งที่สวยงามดูโฉบเฉี่ยว การใช้สไตล์borderRadiusช่วยให้คุณ
ความสามารถในการเพิ่มสไตล์ให้กับแอปพลิเคชันของคุณ คุณสามารถสร้างความแตกต่างระหว่าง

esting รูปร่างโดยการเพิ่มส่วนโค้งในจุดที่ถูกต้อง
ด้วยborderRadiusคุณสามารถกำหนดว่ามุมขอบโค้งมนปรากฏบนองค์ประกอบอย่างไร
เมนชั่น อย่างที่คุณอาจสงสัยborderRadiusใช้กับส่วนประกอบทั้งหมด หากคุณตั้งค่า
borderRadius และไม่ได้ตั้งค่าใดค่าหนึ่งที่เฉพาะเจาะจงมากขึ้นเช่น borderTopLeftRadius ,
มุมทั้งสี่จะถูกปัดเศษ ดูรูป 4.6 เพื่อดูวิธีการปัดเศษเส้นขอบต่างๆ
เพื่อสร้างเอฟเฟกต์สุดเจ๋ง
ตัวอย่างที่ 1:
สี่เหลี่ยมจัตุรัสที่มีมุมมนสี่มุม ตัวอย่าง
มุม
ตัวอย่างที่ 3:
รูปร่างใบ
ตัวอย่างที่ 4:
วงกลม
ตัวอย่างที่ 2:
รูปร่าง D
รูปที่ 4.6 ตัวอย่างการรวมรัศมีเส้นขอบต่างๆ
ตัวอย่างที่ 1: สี่เหลี่ยมจัตุรัสที่มีมุมมนสี่มุม ตัวอย่าง
2: สี่เหลี่ยมจัตุรัสที่มีมุมขวาสองมุมโค้งมนทำให้
รูปร่างดี ตัวอย่างที่ 3: สี่เหลี่ยมจัตุรัสที่มีมุมตรงข้าม
มีลักษณะโค้งมนคล้ายใบไม้ ตัวอย่างที่ 4: สี่เหลี่ยมจัตุรัสที่มี a
รัศมีเส้นขอบเท่ากับครึ่งหนึ่งของความยาวของด้าน ซึ่งส่งผลให้
ในวงกลม
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 17

89
ส่วนประกอบมุมมองจัดแต่งทรงผม
การสร้างรูปร่างในรูปที่ 4.6 นั้นค่อนข้างง่าย ดังที่แสดงในรายการ 4.9 อย่างจริงใจ,
ส่วนที่ยากที่สุดเกี่ยวกับโค้ดนี้คือต้องแน่ใจว่าคุณไม่ได้ทำให้ข้อความใหญ่เกินไปหรือเกินไป
ยาว. ฉันจะแสดงให้คุณเห็นว่าฉันหมายถึงอะไรในไม่ช้าในรายการ 4.10

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { StyleSheet, Text, View} จาก 'react-native';
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<รูปแบบตัวอย่าง={{borderRadius: 20}}>
```

<ข้อความตรงกลาง>
ตัวอย่างที่ 1:{"\n"}4 มุมโค้งมน
</CenteredText>
</ตัวอย่าง>
<รูปแบบตัวอย่าง={{borderTopRightRadius: 60,
borderBottomRightRadius: 60}}>
<ข้อความตรงกลาง>
ตัวอย่าง 2:{"\n"}D รูปร่าง
</CenteredText>
</ตัวอย่าง>
<รูปแบบตัวอย่าง={{borderTopLeftRadius: 30,
borderBottomRightRadius: 30}}>
<ข้อความตรงกลาง>
ตัวอย่างที่ 3:{"\n"}รูปทรงใบไม้
</CenteredText>
</ตัวอย่าง>
<รูปแบบตัวอย่าง={{borderRadius: 60}}>
<ข้อความตรงกลาง>
ตัวอย่างที่ 4:{"\n"}วงกลม
</CenteredText>
</ตัวอย่าง>
</ดู>
);
}
}
const ตัวอย่าง = (อุปกรณ์ประกอบฉาก) => (
<ดู style={[styles.example,props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</ดู>
);
const CenteredText = (อุปกรณ์ประกอบฉาก) => (
<Text style={[styles.centeredText, props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</Text>
);
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ดิ้น: 1,
flexDirection: 'แถว',
flexWrap: 'ห่อ'
ตัวอย่างที่ 1: สี่เหลี่ยมจัตุรัสกับ
สี่มุมโค้งมน
นี่คือจาวาสคริปต์ ดังนั้น
คุณสามารถระบุฮาร์ด

กลับอินไลน์กับ
ข้อความโดยใช้ {"\n"}
ตัวอย่างที่ 2: สี่เหลี่ยมจัตุรัสที่มี
มุมขวาสองโค้งมน
ตัวอย่างที่ 3: สี่เหลี่ยมจัตุรัสที่มี
มุมตรงข้ามโค้งมน
ตัวอย่างที่ 4: สี่เหลี่ยมจัตุรัสที่มีเส้นขอบ
รัศมีเท่ากับครึ่งหนึ่งของความยาวของด้าน
ส่วนประกอบที่ใช้ซ้ำได้
สำหรับการแสดงผล
องค์ประกอบข้อความตรงกลาง
React Native ใช้ flexbox
เพื่อควบคุมการจัดวาง
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 18


```
ขอบด้านบน: 75
},
ตัวอย่าง: {
ความกว้าง: 120,
ความสูง: 120,
ระยะขอบซ้าย: 20,
ขอบด้านล่าง: 20,
พื้นหลังสี: 'สีเทา',
ขอบกว้าง: 2,
justifyContent: 'ศูนย์กลาง'
},
ข้อความกึ่งกลาง: {
textAlign: 'ศูนย์',
ระยะขอบ: 10
}
});
```

ให้ความสนใจเป็นพิเศษกับสไตล์ที่เน้นข้อความ คุณโชคดีโดยใช้
ระยะขอบ: 10 . หากคุณใช้ช่องว่างภายใน: 10 พื้นหลังขององค์ประกอบข้อความจะ

ปิดเส้นขอบขององค์ประกอบมุมมอง (ดูรูปที่ 4.7)
โดยค่าเริ่มต้นคอมโพเนนต์Text จะรับค่าสีพื้นหลังของคอมพาเรนต์
ponent เนื่องจากกรอบขอบของคอมโพเนนต์Textเป็นรูปสี่เหลี่ยมผืนผ้า
พื้นดินทับซ้อนกับมุมโค้งมนที่สวยงาม เห็นได้ชัดว่าการใช้คุณสมบัติระยะขอบช่วยแก้ได้
ปัญหา แต่ก็ยังสามารถแก้ไขสถานการณ์ด้วยวิธีอื่นได้ คุณสามารถเพิ่ม
BackgroundColor: 'โปร่งใส' ไป centeredText สไตล์ การทำข้อความประกอบ-
พื้นหลังโปร่งใสของ nent ช่วยให้เส้นขอบด้านล่างสามารถแสดงผ่านและดูได้
ปกติอีกครั้งดังในรูปที่ 4.6

A DDING BORDERS TO YOUR P ROFILE C ARD COMPONENT

ด้วยความรู้ที่คุณเพิ่งค้นพบเกี่ยวกับคุณสมบัติของเส้นขอบ คุณเกือบจะทำ
เลย์เอาต์เริ่มต้นขององค์ประกอบการ์ดโปรไฟล์ ใช้เฉพาะคุณสมบัติเส้นขอบจาก
ส่วนสุดท้าย คุณสามารถเปลี่ยนสี่เหลี่ยมสี 300 × 400 เป็นสิ่งที่
ใกล้เคียงกับสิ่งที่คุณต้องการมากขึ้น รูปที่ 4.8 แสดงว่าคุณสามารถไปได้ไกลแค่ไหนกับ an
ภาพและเทคนิคที่คุณได้เรียนรู้จนถึงตอนนี้ รวมถึงภาพเพื่อใช้เป็นสถานที่-
ผู้ถือรูปถ่ายของบุคคล คุณจะพบมันในซอร์สโค้ด แต่วงกลมถูกสร้างขึ้น
โดยจัดการรัศมีเส้นขอบตามที่อธิบายไว้ในตัวอย่างก่อนหน้านี้
สไตล์ที่เน้นข้อความ
ภายในองค์ประกอบข้อความ
ตัวอย่างที่ 1:
สี่เหลี่ยมจัตุรัสโค้งมน
มุม
ตัวอย่างที่ 3:
รูปร่างใบ
ตัวอย่างที่ 4:
วงกลม
ตัวอย่างที่ 2:
รูปร่าง D
รูปที่ 4.7 นี่คือหน้าตาของรูปที่ 4.6 ถ้า
รูปแบบข้อความกึ่งกลางที่ใช้ช่องว่างภายใน: 10 แทน
ของระยะขอบ: 10 เพื่อจัดตำแหน่งข้อความ วงกลมเล็กๆ
เน้นจุดที่กรอบของ Text
องค์ประกอบคาบเกี่ยวกับเส้นขอบขององค์ประกอบมุมมอง
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

ส่วนประกอบมุมมองจัดแต่งทรงผม
เห็นได้ชัดว่ามีปัญหาเกี่ยวกับเลย์เอาต์บางอย่างในการ์ดโปรไฟล์ แต่คุณเกือบจะอยู่ที่นั่นแล้ว
เราจะพูดถึงวิธีใช้รูปแบบระยะขอบและช่องว่างภายในในส่วนถัดไปเพื่อรับทุก-
สิ่งที่จัดชิดอย่างถูกต้อง

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { Image, StyleSheet, View} จาก 'react-native';
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<ดู style={styles.cardContainer}>
<ดู style={styles.cardImageContainer}>
<รูปแบบรูปภาพ={styles.cardImage}
source={require('./user.png')}/>
</ดู>
</ดู>
</ดู>
);
}
}
const profileCardColor = 'dodgerblue';
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ดิ้น: 1,
justifyContent: 'ศูนย์',
alignItems: 'ศูนย์'
},
คอนเทนเนอร์การ์ด: {
borderColor: 'สีดำ',
ขอบกว้าง: 3,
borderStyle: 'ของแข็ง',
```

นำเข้ารูปภาพ
ส่วนประกอบจาก
ตอบสนองพื้นเมือง
user.png ตั้งอยู่ในเดียวกัน
ไดเร็กทอรีเป็นรหัสแอปพลิเคชัน
เพิ่มคุณสมบัติเส้นขอบ
ไปที่การ์ดโปรไฟล์
รูปที่ 4.8 การรวมคุณสมบัติเส้นขอบเข้ากับโปรไฟล์
ส่วนประกอบการ์ดแปลงสี่เหลี่ยมสี 300 × 400

เป็นสิ่งที่คล้ายกับสิ่งที่คุณต้องการสำหรับโปรไฟล์สุดท้าย
ส่วนประกอบของการ์ด
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 20


```
รัศมีชายแดน: 20,
พื้นหลังสี: profileCardColor,
ความกว้าง: 300,
ส่วนสูง: 400
},
cardImageContainer: {
พื้นหลังสี: 'สีขาว',
ขอบกว้าง: 3,
borderColor: 'สีดำ',
ความกว้าง: 120,
ความสูง: 120,
รัศมีขอบ: 60,
},
ภาพการ์ด: {
ความกว้าง: 80,
ส่วนสูง: 80
}
});
```

ความแตกต่างระหว่างรายการ 4.10 และรหัสบัตรโปรไฟล์ก่อนหน้า (รายการ 4.7)
ได้เป็นตัวหนาเพื่อเน้นการเปลี่ยนแปลงที่เพิ่มขึ้น

4.2.3 การระบุระยะขอบและช่องว่างภายใน

คุณสามารถวางตำแหน่งทุกองค์ประกอบได้อย่างชัดเจนบน
หน้าจอและจัดวางได้ตามที่คุณต้องการ แต่
ที่จะน่าเบื่ออย่างยิ่งหากจำเป็นต้องวางเลย์เอาต์
เพื่อตอบสนองต่อการกระทำของผู้ใช้ มันสมเหตุสมผลกว่า
เพื่อจัดตำแหน่งสิ่งของให้สัมพันธ์กัน ดังนั้นหากคุณ
ย้ายองค์ประกอบหนึ่งส่วนประกอบอื่น ๆ สามารถ
ย้ายในการตอบสนองตามตำแหน่งที่สัมพันธ์กัน
รูปแบบระยะขอบช่วยให้คุณสามารถกำหนดความสัมพันธ์นี้-
จัดส่งระหว่างส่วนประกอบ รูปแบบช่องว่างภายในช่วยให้

คุณกำหนดตำแหน่งสัมพัทธ์ของส่วนประกอบเป็น
ชายแดนของมัน การใช้คุณสมบัติเหล่านี้ร่วมกันทำให้ a
มีความยืดหยุ่นสูงในการจัดวางส่วนประกอบ
คุณจะใช้คุณสมบัติเหล่านี้ทุกวันจึงสำคัญ
เพื่อทำความเข้าใจสิ่งที่พวกเขาหมายถึงและทำ
แนวความคิด ระยะขอบและช่องว่างภายในทำงานอย่างแน่นอน
เช่นเดียวกับที่ทำใน CSS การพรรณนาตามธรรมเนียมของระยะขอบและช่องว่างภายใน
เกี่ยวข้องกับเส้นขอบและพื้นที่เนื้อหายังคงมีผลบังคับใช้ (ดูรูปที่ 4.9)
ในทางปฏิบัติ คุณอาจพบจุดบกพร่องเมื่อต้องรับมือกับระยะขอบและช่องว่างภายใน
คุณอาจถูกล่อลวงให้เรียกพวกเขาว่า "นิสัยใจคอ" แต่ไม่ว่าอย่างไรมันก็เจ็บปวด มากที่สุด
ส่วนระยะขอบบนองค์ประกอบการดูทำงานได้ดีพอสมควรและทำงานบน iOS และ
แอนดรอยด์ Padding มีแนวโน้มที่จะทำงานแตกต่างกันเล็กน้อยระหว่าง OS ในขณะที่เขียน
ส่วนประกอบข้อความแพ็ดในสภาพแวดล้อม Android ไม่ทำงานเลย ฉันสงสัยว่า
จะมีการเปลี่ยนแปลงในรุ่นถัดไป
คอนเทนเนอร์รูปภาพมีขนาด 120 × 120 square
มีเส้นขอบรัศมี 60 (ครึ่งหนึ่งของ 120) ซึ่ง
ส่งผลให้เป็นวงกลม
สไตล์สำหรับภาพจริง
มาร์จิ้น
ชายแดน
การขยายความ
ส่วนประกอบ
รูปที่ 4.9 การพรรณนาทั่วไปของ
ระยะขอบ ช่องว่างภายใน และเส้นขอบอย่างไร
สัมพันธ์กัน
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 21

93
ส่วนประกอบมุมมองจัดแต่งทรงผม
คุณร้องเพลงคุณสมบัติ Margin
เมื่อจัดวางส่วนประกอบ ปัญหาแรกที่ต้องแก้ไขอย่างหนึ่งก็คือว่าคอม
ponents มาจากกันและกัน เพื่อหลีกเลี่ยงการระบุระยะทางสำหรับแต่ละองค์ประกอบ
คุณต้องมีวิธีระบุตำแหน่งสัมพัทธ์ ขอบคุณสมบัติช่วยให้คุณกำหนด

เส้นรอบวงของส่วนประกอบ ซึ่งกำหนดว่าองค์ประกอบอยู่ห่างจาก
องค์ประกอบก่อนหน้าหรือหลัก การวางเลย์เอาต์ด้วยวิธีนี้ทำให้คอนเทนเนอร์
เพื่อหาว่าส่วนประกอบควรอยู่ในตำแหน่งใดโดยสัมพันธ์กัน
แทนที่จะต้องคำนวณตำแหน่งของทุกองค์ประกอบ
คุณสมบัติมาร์จิ้นที่ใช้ได้คือmargin , marginTop , marginRight , margin-
ล่างและmarginLeft หากมีการตั้งค่าคุณสมบัติระยะขอบทั่วไปเท่านั้นโดยไม่มีการตั้งค่าอื่น
ค่าเฉพาะเจาะจงมากขึ้น เช่นmarginLeftหรือmarginTopจากนั้นค่านั้นจะมีผลกับทั้งหมด
ด้านข้างของส่วนประกอบ (บน ขวา ล่าง และซ้าย) ถ้าทั้งระยะขอบและความเร็วที่มากขึ้น-
มีการระบุคุณสมบัติระยะขอบ cific (เช่นmarginLeft) จากนั้นระบุคุณสมบัติเพิ่มเติม
ขอบอสังหาริมทรัพย์จะมีความสำคัญ มันทำงานเหมือนกับคุณสมบัติของเส้นขอบ
ลองใช้รูปแบบเหล่านี้ดู รูปภาพ 4.10
ระยะขอบทั้งหมดวางตำแหน่งส่วนประกอบตามที่คาดไว้ แต่สังเกตว่า Android
อุปกรณ์คลิปส่วนประกอบเมื่อใช้ระยะขอบติดลบ หากคุณวางแผนที่จะสนับสนุน
ทั้ง iOS และ Android ทดสอบบนอุปกรณ์แต่ละเครื่องตั้งแต่เริ่มต้นโครงการของคุณ อย่า
พัฒนาบน iOS และคิดว่าทุกสิ่งที่คุณจัดสไตล์จะมีลักษณะเหมือนกันบน Android รายการ-
ing 4.11 แสดงรหัสสำหรับตัวอย่างในรูปที่ 4.10

นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { StyleSheet, Text, View} จาก 'react-native';
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
NS
iOS
Android
ไม่มีมาร์จิ้น
NS
marginTop
marginLeft
เชิงลบ
ขอบด้านบน &
marginLeft
การตัดขอบติดลบ
marginTop
ค
NS
ค
NS
NS
NS
รูปที่ 4.10 ตัวอย่างการใช้ระยะขอบกับส่วนประกอบ ใน iOS ตัวอย่าง A ไม่มีระยะขอบ
สมัครแล้ว. ตัวอย่าง B ใช้ระยะขอบบน ตัวอย่าง C มีระยะขอบด้านบนและด้านซ้าย ตัวอย่าง D มี
ทั้งระยะขอบซ้ายบนและติดลบ ใน Android ระยะขอบติดลบมีพฤติกรรมแตกต่างไปเล็กน้อย:
คอมโพเนนต์ถูกตัดโดยคอนเทนเนอร์หลัก

หน้า 22


```
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<ดู style={styles.exampleContainer}>
<ตัวอย่าง>
<CenteredText>A</CenteredText>
</ตัวอย่าง>
</ดู>
<ดู style={styles.exampleContainer}>
<รูปแบบตัวอย่าง={{ขอบบน: 50}}>
<CenteredText>B</CenteredText>
</ตัวอย่าง>
</ดู>
<ดู style={styles.exampleContainer}>
<รูปแบบตัวอย่าง={{marginTop: 50, marginLeft: 10}}>
<CenteredText>C</CenteredText>
</ตัวอย่าง>
</ดู>
<ดู style={styles.exampleContainer}>
<รูปแบบตัวอย่าง={{marginLeft: -10, marginTop: -10}}>
<CenteredText>D</CenteredText>
</ตัวอย่าง>
</ดู>
</ดู>
);
}
}
const ตัวอย่าง = (อุปกรณ์ประกอบฉาก) => (
<ดู style={[styles.example,props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</ดู>
);
const CenteredText = (อุปกรณ์ประกอบฉาก) => (
<Text style={[styles.centeredText, props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</Text>
);
```

รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
alignItems: 'ศูนย์',
ดิ้น: 1,
flexDirection: 'แถว',
flexWrap: 'ห่อ'
justifyContent: 'ศูนย์',
ขอบด้านบน: 75
},
ตัวอย่างคอนเทนเนอร์: {
ขอบกว้าง: 1,
ความกว้าง: 120,
ความสูง: 120,
ระยะขอบซ้าย: 20,
ขอบด้านล่าง: 20,
ตัวอย่างฐานด้วย
ไม่มีการใช้ระยะขอบ
ขอบด้านบนของ50
ขอบด้านบนของ 50 และ
marginLeft of 10
ใช้ระยะขอบติดลบกับ
marginTop และ marginLeft
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 23

95
ส่วนประกอบมุมมองจัดแต่งทรงผม
},
ตัวอย่าง: {
ความกว้าง: 50,
ความสูง: 50,
พื้นหลังสี: 'สีเทา',
ขอบกว้าง: 1,
justifyContent: 'ศูนย์กลาง'
},
ข้อความกึ่งกลาง: {
textAlign: 'ศูนย์',
ระยะขอบ: 10
}

});

คุณร้องเพลงพรีอพเพอร์ตี้

คุณสามารถคิดว่าระยะขอบเป็นระยะห่างระหว่างองค์ประกอบ แต่ช่องว่างภายในแสดงถึง
ช่องว่างระหว่างเนื้อหาขององค์ประกอบและเส้นขอบขององค์ประกอบเดียวกัน เมื่อไหร่
มีการระบุช่องว่างภายใน จะช่วยให้เนื้อหาขององค์ประกอบไม่ถูกล้างออกกับ
ชายแดน. ในรูปที่ 4.9 คุณสมบัติbackgroundColorตกผ่านส่วนประกอบ
ขอบถึงชายแดนซึ่งเป็นพื้นที่ที่กำหนดโดยpadding ที่เหมาะสมที่มีอยู่ -
เนคไทที่ใช้ได้สำหรับpaddingคือpadding , paddingLeft , paddingRight , paddingTop , และ
paddingBottom . หากตั้งค่าคุณสมบัติช่องว่างภายในหลักเท่านั้นโดยไม่มีการตั้งค่าอื่นที่เฉพาะเจาะจงมากขึ้น
ค่าเช่นpaddingLeftหรือpaddingTopจากนั้นค่านั้นจะถูกส่งต่อไปยังทุกด้านของ
ส่วนประกอบ (บน ขวา ล่าง และซ้าย) หากทั้งสองช่องว่างภายในและอื่น ๆ เฉพาะpadding
มีการระบุคุณสมบัติ เช่นpaddingLeftจากนั้นคุณสมบัติpadding ที่เฉพาะเจาะจงมากขึ้น
มีความสำคัญ ลักษณะการทำงานนี้เหมือนกับเส้นขอบและระยะขอบ
แทนที่จะสร้างตัวอย่างใหม่เพื่อแสดงให้เห็นว่าช่องว่างภายในแตกต่างจากระยะขอบอย่างไร
มานำโค้ดจากรายการ 4.11 กลับมาใช้ใหม่และปรับแต่งเล็กน้อย เปลี่ยนรูปแบบระยะขอบ
ส่วนประกอบเช่นการขยายรูปแบบและเพิ่มเส้นขอบรอบข้อความซับซ้อน
ponents และเปลี่ยนสีพื้นหลังของพวกเขา รูปที่ 4.11 แสดงสิ่งที่คุณจะลงเอยด้วย

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
...
<ดู style={styles.container}>
<ดู style={styles.exampleContainer}>
<รูปแบบตัวอย่าง={{}}>
<CenteredText>A</CenteredText>
</ตัวอย่าง>
</ดู>
<ดู style={styles.exampleContainer}>
<รูปแบบตัวอย่าง={{paddingTop: 50}}>
<CenteredText>B</CenteredText>
</ตัวอย่าง>
</ดู>
<ดู style={styles.exampleContainer}>
<รูปแบบตัวอย่าง={{paddingTop: 50, paddingLeft: 10}}>
<CenteredText>C</CenteredText>
```

ตัวอย่าง A: ไม่เปลี่ยนแปลง กับ
ไม่มีขอบหรือช่องว่างภายใน
ตัวอย่าง B: marginTop
เปลี่ยนเป็น paddingTop

ตัวอย่าง C: marginTop และ marginLeft เปลี่ยนไป
เพื่อ paddingTop และ paddingLeft ตามลำดับ
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 24


```
</ตัวอย่าง>
</ดู>
<ดู style={styles.exampleContainer}>
<รูปแบบตัวอย่าง={{paddingLeft: -10, paddingTop: -10}}>
<CenteredText>D</CenteredText>
</ตัวอย่าง>
</ดู>
</ดู>
...
},
ข้อความกึ่งกลาง: {
textAlign: 'ศูนย์',
ระยะขอบ: 10,
ขอบกว้าง: 1,
พื้นหลังสี: 'สีเทาอ่อน'
}
});
```

ไม่เหมือนกับระยะขอบ ซึ่งระบุช่องว่างระหว่างส่วนประกอบและองค์ประกอบหลัก nent การเติมจะใช้จากขอบของส่วนประกอบไปยังลูกของมัน ในตัวอย่าง B ช่องว่างภายในคำนวณจากเส้นขอบด้านบนซึ่งผลักองค์ประกอบข้อความ B ลง จากขอบด้านบน ตัวอย่าง C เพิ่มค่าpaddingLeftซึ่งจะผลักText องค์ประกอบ C เข้าด้านในจากขอบด้านซ้าย ตัวอย่าง D ใช้ค่าช่องว่างภายในเชิงลบ เพื่อpaddingTopและpaddingLeft
ตัวอย่าง D: marginLeft และ marginTop
เปลี่ยนเป็น paddingLeft และ marginTop
ตามลำดับ ค่าลบยังคงอยู่
เพิ่มเส้นขอบและพื้นหลัง
สีให้กับองค์ประกอบข้อความ

paddingLeft
เชิงลบ
paddingด้านบน &
paddingLeft
การตัดแพดดิ้ง
paddingTop
Android
NS
NS
ค
NS
NS
ค
การขยายความ
คลิปหนีบ
NS
NS
รูปที่ 4.11 การเปลี่ยนรูปแบบระยะขอบจากตัวอย่างก่อนหน้าเป็นรูปแบบการเติม ตัวอย่าง A,
ไม่มีช่องว่างภายใน จะดูเหมือนกับเมื่อไม่มีการใช้ระยะขอบ ตัวอย่าง B แสดงส่วนประกอบ
ด้วย paddingTop ถูกนำไปใช้ ตัวอย่าง C เหมือนกัน แต่ใช้ paddingLeft ด้วย ตัวอย่าง D
ใช้ค่าช่องว่างภายในเชิงลบกับ paddingTop และ paddingLeft ซึ่งจะถูกละเว้น
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 25

97
ส่วนประกอบมุมมองจัดแต่งทรงผม
ข้อสังเกตที่น่าสนใจบางประการ ตัวอย่าง B และตัวอย่าง C เป็นทั้งคู่
ตัดบนอุปกรณ์ Android ความกว้างขององค์ประกอบข้อความของตัวอย่าง C ถูกบีบอัด
และค่าลบสำหรับการเติมจะถูกละเว้นในตัวอย่าง D

4.2.4 การใช้ตำแหน่งการวางส่วนประกอบ

จนถึงตอนนี้ ทุกสิ่งที่เราดูมีการจัดตำแหน่งที่สัมพันธ์กับองค์ประกอบอื่น
ซึ่งเป็นตำแหน่งเค้าโครงเริ่มต้น บางครั้งก็มีประโยชน์ที่จะใช้ประโยชน์จาก
การวางตำแหน่งที่แน่นอนและวางส่วนประกอบในตำแหน่งที่คุณต้องการ ผู้ประกอบการ-
tation ของสไตล์ตำแหน่งใน React Native นั้นคล้ายกับ CSS แต่มีไม่มากนัก
ตัวเลือก. โดยค่าเริ่มต้น องค์ประกอบทั้งหมดจะถูกจัดวางโดยสัมพันธ์กัน หากมีการกำหนดตำแหน่ง
เป็นAbsoluteจากนั้นองค์ประกอบจะถูกจัดวางสัมพันธ์กับพาเรนต์ คุณสมบัติที่มีอยู่
สำหรับตำแหน่งที่มีญาติ (ตำแหน่งเริ่มต้น) และแน่นอน
CSS มีค่าอื่น ๆ แต่นั่นเป็นเพียงสองค่าใน React Native เมื่อใช้
ตำแหน่งที่แน่นอนคุณสมบัติดังต่อไปนี้ยังมีอยู่: top , right , bot-

ทอมและจากไป
มาดูตัวอย่างง่ายๆ เพื่อแสดงความแตกต่างระหว่างญาติกับ
ตำแหน่งที่แน่นอน ใน CSS การวางตำแหน่งอาจทำให้สับสนมากขึ้น แต่ใน React
เนทีฟ "ทุกอย่างมีตำแหน่งสัมพัทธ์โดยค่าเริ่มต้น" ทำให้ง่ายต่อการ
รายการตำแหน่ง ในรูปที่ 4.12 บล็อก A, B และ C ถูกจัดวางสัมพันธ์กันใน
แถว. ไม่มีขอบหรือช่องว่างภายใน พวกมันจะเรียงต่อกัน Block D คือ a
พี่น้องกับแถว ABC ของบล็อก หมายความว่าคอนเทนเนอร์หลักคือคอนเทนเนอร์หลัก
สำหรับแถว ABC และบล็อก D
Block D ถูกตั้งค่าเป็น {position: 'absolute', right: 0, bottom: 0} ดังนั้นจึงอยู่ในตำแหน่ง
ที่มุมล่างขวาของคอนเทนเนอร์ บล็อก E ถูกตั้งค่าเป็น {ตำแหน่ง: 'สัมบูรณ์'
right: 0, bottom: 0} แต่คอนเทนเนอร์หลักคือ block B ซึ่งหมายความว่า block E เป็น posi-
ระบุไว้อย่างแน่นอน แต่เกี่ยวกับบล็อก B บล็อก E ปรากฏขึ้นที่มุมล่างขวา
ของบล็อก B แทน รายการ 4.13 แสดงรหัสสำหรับตัวอย่างนี้
NS
NS
ค
NS
อี
รูปที่ 4.12 ตัวอย่างแสดงการวางบล็อก A, B และ C
สัมพันธ์กัน. Block D มีตำแหน่งที่แน่นอนของสิทธิ:
0 และด้านล่าง: 0 บล็อก E มีตำแหน่งที่แน่นอนเช่นกัน: 0
และด้านล่าง: 0 แต่พาเรนต์คือบล็อก B ไม่ใช่คอนเทนเนอร์หลัก
ในขณะที่พาเรนต์ของ D เป็นคอนเทนเนอร์หลัก
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 26


```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { StyleSheet, Text, View} จาก 'react-native';
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<ดู style={styles.row}>
<ตัวอย่าง>
```

<CenteredText>A</CenteredText>
</ตัวอย่าง>
<ตัวอย่าง>
<CenteredText>B</CenteredText>
<ดูสไตล์={[styles.tinyExample,
{ตำแหน่ง: 'แน่นอน',
ขวา: 0,
ด้านล่าง: 0}]}>
<CenteredText>E</CenteredText>
</ดู>
</ตัวอย่าง>
<ตัวอย่าง>
<CenteredText>C</CenteredText>
</ตัวอย่าง>
</ดู>
<รูปแบบตัวอย่าง={{ตำแหน่ง: 'แน่นอน',
ขวา: 0, ล่าง: 0}}>
<CenteredText>D</CenteredText>
</ตัวอย่าง>
</ดู>
);
}
}
const ตัวอย่าง = (อุปกรณ์ประกอบฉาก) => (
<ดู style={[styles.example,props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</ดู>
);
const CenteredText = (อุปกรณ์ประกอบฉาก) => (
<Text style={[styles.centeredText, props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</Text>
);
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ความกว้าง: 300,
ความสูง: 300,
ระยะขอบ: 40,
ขอบบน: 100,
borderWidth: 1
},
แถว: {
แถวที่มีบล็อก A, B และ C
บล็อคอีแน่นอน

อยู่ใน
มุมล่างขวา
ของผู้ปกครอง
ตู้คอนเทนเนอร์ บล็อคบี
บล็อคดีแน่นอน
อยู่ใน
มุมล่างขวาของ
คอนเทนเนอร์หลัก
ทิศทางของ flexbox ถูกระบุเป็นแถว ดังนั้น
บล็อกอยู่ในแถวข้ามหน้าจอ
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 27

99
ส่วนประกอบมุมมองจัดแต่งทรงผม
ดิ้น: 1,
flexDirection: 'แถว'
},
ตัวอย่าง: {
ความกว้าง: 100,
ความสูง: 100,
พื้นหลังสี: 'สีเทา',
ขอบกว้าง: 1,
justifyContent: 'ศูนย์กลาง'
},
tinyตัวอย่าง: {
ความกว้าง: 30,
ความสูง: 30,
ขอบกว้าง: 1,
justifyContent: 'ศูนย์',
พื้นหลังสี: 'สีเทาอ่อน'
},
ข้อความกึ่งกลาง: {
textAlign: 'ศูนย์',
ระยะขอบ: 10
}
});
หมายเหตุในรายการ 4.13 คุณสมบัติflexDirection ถูกระบุเป็น'row' ดังนั้น

บล็อกอยู่ในแถวข้ามหน้าจอ React Native ใช้โอเพ่นซอร์ส
ไลบรารีเลย์เอาต์ข้ามแพลตฟอร์มที่เรียกว่า Yoga (https://yogalayout.com) โยคะ
ใช้โหมดเลย์เอาต์ flexbox ซึ่งคุณมักจะเห็นใน CSS และถูกใช้
บ่อยครั้งใน React Native เราจะใช้เวลามากมายในบทต่อไปในการพูดคุย
เกี่ยวกับเฟล็กซ์บ็อกซ์ ระยะขอบ ช่องว่างภายใน และตำแหน่งเป็นเครื่องมือการจัดวางที่ยอดเยี่ยม แต่
flexbox เป็นเครื่องมือที่คุณจะใช้บ่อยที่สุด
จบด้วยพื้นฐานของการจัดสไตล์ส่วนประกอบViewแล้ว คุณได้เรียนรู้
เกี่ยวกับเทคนิคการจัดวางบางอย่าง: ระยะขอบ ช่องว่างภายใน และตำแหน่ง มาทบทวนโปรไฟล์กันเถอะ
ส่วนประกอบการ์ดและแก้ไขชิ้นส่วนที่ยังไม่ได้จัดวางอย่างเหมาะสม

4.2.5 การวางตำแหน่งโปรไฟล์การ์ด

รายการต่อไปนี้มีการเปลี่ยนแปลงรหัสที่ต้องทำกับรายการ 4.10 ไปยัง space
วงกลมและภาพผู้ใช้อย่างถูกต้องและจัดกึ่งกลางทุกอย่าง รูปที่ 4.13 แสดงผล

```
...
คอนเทนเนอร์การ์ด: {
alignItems: 'ศูนย์',
borderColor: 'สีดำ',
ขอบกว้าง: 3,
borderStyle: 'ของแข็ง',
รัศมีชายแดน: 20,
พื้นหลังสี: profileCardColor,
ความกว้าง: 300,
ส่วนสูง: 400
จัดแนววงกลมในแนวนอน
ศูนย์กลางของบัตรโปรไฟล์
```

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

```
},
cardImageContainer: {
alignItems: 'ศูนย์',
พื้นหลังสี: 'สีขาว',
ขอบกว้าง: 3,
borderColor: 'สีดำ',
```

ความกว้าง: 120,
ความสูง: 120,
รัศมีขอบ: 60,
ขอบบน: 30,
paddingTop: 15
},
...
ตอนนี้ส่วนประกอบมุมมองหลักสำหรับการ์ดโปรไฟล์อยู่ในตำแหน่งแล้ว โดยใช้เทคโนโลยี-
คุณได้สร้างรากฐานที่สวยงามสำหรับส่วนประกอบ แต่
คุณยังไม่เสร็จ คุณต้องเพิ่มข้อมูลเกี่ยวกับบุคคล: ชื่อ, อาชีพ,
และคำอธิบายโปรไฟล์โดยย่อ ข้อมูลทั้งหมดนั้นเป็นข้อความดังนั้นสิ่งต่อไป
คุณจะได้เรียนรู้วิธีจัดรูปแบบองค์ประกอบข้อความ

4.3 องค์ประกอบการจัดรูปแบบข้อความ

ในส่วนนี้ เราจะพูดถึงวิธีจัดรูปแบบองค์ประกอบข้อความ หลังเลิกงาน
ความรู้เกี่ยวกับวิธีการทำให้ข้อความดูดีเราจะดูที่การ์ดโปรไฟล์
และเพิ่มข้อมูลบางอย่างเกี่ยวกับผู้ใช้ รูปที่ 4.14 เป็นคอมฯ Profile Card ที่เสร็จแล้ว
ponent พร้อมชื่อผู้ใช้และอาชีพและคำอธิบายโปรไฟล์โดยย่อ แต่ก่อน
เรากลับมาดูโปรไฟล์การ์ด มาดูเทคนิคการจัดแต่งทรงผมที่จะช่วยให้คุณ
เสร็จสิ้นการสร้างมัน

4.3.1 ส่วนประกอบข้อความเทียบกับดูส่วนประกอบ

ยกเว้นคุณสมบัติ flex ซึ่งเรายังไม่ครอบคลุม รูปแบบส่วนใหญ่
ที่ใช้กับองค์ประกอบการดูจะทำงานตามที่คาดไว้กับองค์ประกอบข้อความ ข้อความ
จัดแนวภาพผู้ใช้ใน
ศูนย์กลางแนวนอนของวงกลม
ให้ช่องว่างระหว่างส่วนบนของ
วงกลมและด้านบนของโปรไฟล์การ์ด
ให้เบาะรองนั่ง
ระหว่างภายใน
ส่วนหนึ่งของวงกลม
และที่บรรจุ
ภาพ
รูปที่ 4.13 ส่วนประกอบ Profile Card หลังจาก View . ทั้งหมด
ส่วนประกอบได้รับการจัดวางอย่างถูกต้อง

101

การจัดองค์ประกอบข้อความ

องค์ประกอบสามารถมีเส้นขอบและพื้นหลัง และได้รับผลกระทบจากคุณสมบัติของเค้าโครง เช่น
อัตรากำไรขั้นต้น, padding และตำแหน่ง
กลับพูดไม่ได้ สไตล์ส่วนใหญ่องค์ประกอบข้อความที่ใช้ได้จะไม่ทำงาน
สำหรับดูองค์ประกอบ ซึ่งเหมาะสมอย่างยิ่ง หากคุณเคยใช้กระบวนการคำ
คุณรู้ไหมว่าคุณสามารถใช้แบบอักษรต่างๆ สำหรับข้อความและเปลี่ยนสีแบบอักษรได้ นั่น
คุณสามารถปรับขนาด ตัวหนา และตัวเอียงข้อความได้ และคุณสามารถนำการตกแต่งเช่น
ขีดเส้นใต้
ก่อนที่เราจะพูดถึงสไตล์เฉพาะข้อความ เรามาพูดถึงสีกันก่อน สไตล์ที่ใช้กันทั้งคู่
ส่วนประกอบข้อความและมุมมอง จากนั้นคุณจะใช้สีร่วมกับทุกสิ่งที่คุณได้เรียนรู้
จนถึงตอนนี้เพื่อเริ่มเพิ่มข้อความลงในการ์ดโปรไฟล์

C OLORING TEXT

สีคุณสมบัติใช้กับข้อความองค์ประกอบในทางเดียวกันว่ามันไม่
เพื่อดูส่วนประกอบ ตามที่คาดไว้ คุณสมบัตินี้ระบุสีของข้อความใน a
องค์ประกอบข้อความ ทุกรูปแบบสีที่แสดงในตาราง 4.1 ยังคงใช้แม้กระทั่งความโปร่งใส,
แม้ว่าฉันจะนึกไม่ออกว่ามันมีประโยชน์อย่างไร โดยค่าเริ่มต้น สีข้อความจะเป็นสีดำ
รูปที่ 4.14 แสดงองค์ประกอบข้อความสามองค์ประกอบในการ์ดโปรไฟล์:

- ชื่อ
- อาชีพ
- รายละเอียดโปรไฟล์

จากสิ่งที่คุณได้เรียนรู้ไปแล้ว คุณสามารถจัดตำแหน่งและจัดตำแหน่งข้อความ เปลี่ยน
สีของชื่อจากสีดำเป็นสีขาว และเพิ่มเส้นขอบอย่างง่ายเพื่อแยก occu-
จากคำอธิบาย รูปที่ 4.15 แสดงสิ่งที่คุณจะลงเอยด้วยการสมัคร
เทคนิคในคลังแสงของคุณ
โดยจุดนี้ท่านน่าจะทำตามข้อ 4.15 ได้และเข้าใจ
ทุกสิ่งทุกอย่างที่เกิดขึ้น อย่ารู้สึกแย่หากไม่—หากจำเป็น ให้กลับไปอ่านใหม่
ส่วนที่เหมาะสม

React Native Developer
อาชีพ
ชื่อ
John เป็น JavaScript ที่ขอดเขี้ยมจริงๆ
นักพัฒนา เขาชอบใช้ JS to

สร้างแอปพลิเคชัน React Native
สำหรับ iOS และ Android
ประวัติโดยย่อ
รายละเอียดรูปที่ 4.14 กรอกโปรไฟล์การ์ดกับ
ชื่อผู้ใช้และอาชีพและบทสรุป
คำอธิบายโปรไฟล์
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 30


```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { รูปภาพ, สไตล์ชีต, ข้อความ, มุมมอง} จาก 'react-native';
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<ดู style={styles.cardContainer}>
<ดู style={styles.cardImageContainer}>
<รูปแบบรูปภาพ={styles.cardImage}
source={require('./user.png')}/>
</ดู>
<ดู>
<Text style={styles.cardName}>
จอห์น โด
</Text>
</ดู>
<ดู style={styles.cardOccupationContainer}>
<Text style={styles.cardOccupation}>
React Native Developer
</Text>
</ดู>
<ดู>
<Text style={styles.cardDescription}>
John เป็นนักพัฒนา JavaScript ที่ยอดเยี่ยมจริงๆ เขา
ชอบใช้ JS เพื่อสร้างแอปพลิเคชัน React Native
สำหรับ iOS และ Android
</Text>
</ดู>
</ดู>
```

</ดู>
);

React Native Developer
John เป็น JavaScript ที่ยอดเยี่ยมจริงๆ
นักพัฒนา เขาชอบใช้ JS to
สร้างแอปพลิเคชัน React Native
สำหรับ iOS และ Android
รูปที่ 4.15 การ์ดโปรไฟล์ที่เพิ่มองค์ประกอบข้อความ
โดยใช้ค่าเริ่มต้นของการจัดรูปแบบข้อความและคุณสมบัติสีสำหรับ
ตั้งชื่อเป็นสีขาว
นำเข้าส่วนประกอบข้อความ
จาก react-native
องค์ประกอบข้อความ
ที่ทำให้
ชื่อบุคคล
คอนเทนเนอร์รอบ
ข้อความอาชีพที่กำหนด
ขอบล่าง
การแยกอาชีพ
จากคำอธิบาย
องค์ประกอบข้อความที่
ทำให้อาชีพ
องค์ประกอบข้อความที่แสดง
คำอธิบายโปรไฟล์
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 31

103
การจัดองค์ประกอบข้อความ
}
}
const profileCardColor = 'dodgerblue';
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ดิ้น: 1,
justifyContent: 'ศูนย์',

```
alignItems: 'ศูนย์'
},
คอนเทนเนอร์การ์ด: {
alignItems: 'ศูนย์',
borderColor: 'สีดำ',
ขอบกว้าง: 3,
borderStyle: 'ของแข็ง',
รัศมีชายแดน: 20,
พื้นหลังสี: profileCardColor,
ความกว้าง: 300,
ส่วนสูง: 400
},
cardImageContainer: {
alignItems: 'ศูนย์',
พื้นหลังสี: 'สีขาว',
ขอบกว้าง: 3,
borderColor: 'สีดำ',
ความกว้าง: 120,
ความสูง: 120,
รัศมีขอบ: 60,
ขอบบน: 30,
paddingTop: 15
},
ภาพการ์ด: {
ความกว้าง: 80,
ส่วนสูง: 80
},
ชื่อการ์ด: {
สี: 'ขาว',
ขอบบน: 30,
},
การ์ดอาชีพคอนเทนเนอร์: {
borderColor: 'สีดำ',
borderBottomWidth: 3
},
การ์ดอาชีพ: {
ขอบบน: 10,
ระยะขอบด้านล่าง: 10,
},
cardDescription: {
ขอบบน: 10,
ขอบขวา: 40,
ระยะขอบซ้าย: 40,
marginBottom: 10
}
```

});
ลักษณะสำหรับชื่อองค์ประกอบข้อความ
สีคือ 'สีขาว'
รูปแบบสำหรับคอนเทนเนอร์อาชีพ
รูปแบบข้อความอาชีพ (ปัจจุบันคือ
เฉพาะการจัดวางตำแหน่ง)
ลักษณะสำหรับคำอธิบายโปรไฟล์
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 32

ณ จุดนี้ คุณมีเนื้อหาทั้งหมดสำหรับการ์ดโปรไฟล์ แต่ก็ค่อนข้างธรรมดา ใน
ต่อไปเราจะพูดถึงวิธีตั้งค่าคุณสมบัติฟอนต์และเพิ่มการตกแต่ง
รูปแบบข้อความ

4.3.2 รูปแบบตัวอักษร

หากคุณเคยใช้โปรแกรมประมวลผลคำหรือเขียนอีเมลที่มีความสามารถข้อความที่หลากหลาย คุณได้
สามารถเปลี่ยนฟอนต์ เพิ่มหรือลดขนาดฟอนต์ ตัวหนาหรือตัวเอียงได้
และอื่นๆ นี่เป็นสไตล์เดียวกับที่คุณจะได้เรียนรู้วิธีเปลี่ยนแปลงในส่วนนี้ โดยการปรับ-
ด้วยสไตล์เหล่านี้ คุณสามารถทำให้ข้อความดูน่าสนใจและน่าสนใจยิ่งขึ้นสำหรับผู้ใช้ปลายทาง ดี
หารือเกี่ยวกับคุณสมบัติเหล่านี้: fontFamily , FontSize , fontStyleและfontWeight

S PECIFYING ครอบครัวแบบอักษร

fontFamilyทรัพย์สินเรียบง่าย หากคุณยึดติดกับค่าเริ่มต้น ก็ง่าย
แต่ถ้าคุณต้องการใช้แบบอักษรเฉพาะ คุณสามารถประสบปัญหาได้อย่างรวดเร็ว ทั้ง iOS และ
Android มาพร้อมกับชุดแบบอักษรเริ่มต้น สำหรับ iOS ฟอนต์ที่มีอยู่จำนวนมากสามารถ
นำไปปฏิบัตินอกกรอบ สำหรับ Android มี Roboto แบบอักษรโมโนสเปซ และ
ตัวแปร serif และ sans serif อย่างง่าย สำหรับรายการแบบอักษร Android และ iOS ทั้งหมดที่มีให้
ออกจากกล่องใน React พื้นเมืองไปที่https://github.com/dabit3/react-native-fonts
หากคุณต้องการใช้ฟอนต์แบบโมโนสเปซในแอพพลิเคชั่น คุณไม่สามารถระบุได้เช่นกัน
ดังต่อไปนี้:

- fontFamily: 'monospace' — iOS ไม่รองรับตัวเลือก'monospace' ดังนั้น
บนแพลตฟอร์มนั้น คุณจะได้รับข้อผิดพลาด “Unrecognized font family ' monospace ' .”
แต่บน Android ฟอนต์จะแสดงผลอย่างถูกต้องโดยไม่มีปัญหาใดๆ ต่างจาก CSS

คุณไม่สามารถใส่แบบอักษรหลายแบบให้กับคุณสมบัติfontFamily

- fontFamily: 'American Typewriter, monospace' — คุณจะได้รับข้อผิดพลาดอีกครั้ง บน iOS, “ตระกูลฟอนต์ที่ไม่รู้จัก' American Typewriter, monospace ' ” แต่เมื่อ Android เมื่อคุณจัดหาแบบอักษรที่ไม่รองรับ แบบอักษรนั้นจะเปลี่ยนกลับไปเป็นค่าเริ่มต้น นั่นอาจไม่เป็นจริงใน Android ทุกรุ่น แต่ก็เพียงพอแล้วที่จะไม่พูด

วิธีการจะทำงาน

หากคุณต้องการใช้ฟอนต์ที่แตกต่างกัน คุณจะต้องใช้ส่วนประกอบแพลตฟอร์มของ React Native คุยกันเรื่องPlatformอย่างละเอียดในบทที่ 10 แต่ผมอยากแนะนำคุณ ดูวิธีการแก้ไขภาวะที่กลืนไม่เข้าคายไม่ออกนี้ รูปที่ 4.16 แสดงแบบอักษรเครื่องพิมพ์ดีดอเมริกัน แสดงผลบน iOS และแบบอักษร monospace ทั่วไปที่ใช้บน Android รายการต่อไปนี้แสดงรหัสที่สร้างตัวอย่างนี้ ให้ความสนใจกับ วิธีการfontFamilyจะถูกตั้งค่าการใช้Platform.select

รูปที่ 4.16 ตัวอย่างการเรนเดอร์
แบบอักษรโมโนสเปซทั้งบน iOS และ Android

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 33

105
การจัดองค์ประกอบข้อความ

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { Platform, StyleSheet, Text, View} จาก 'react-native';
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<ดู style={styles.row}>
<ข้อความตรงกลาง>
ฉันเป็นแบบอักษรโมโนสเปซบนทั้งสองแพลตฟอร์ม
</CenteredText>
<BottomText>
{Platform.OS}
</BottomText>
</ดู>
</ดู>
);
}
}
```

```
const CenteredText = (อุปกรณ์ประกอบฉาก) => (
<Text style={[styles.centeredText, props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</Text>
);
const BottomText = (อุปกรณ์ประกอบฉาก) => (
<CenteredText style={[{position: 'absolute', bottom: 0},
props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</CenteredText>
);
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ความกว้าง: 300,
ความสูง: 300,
ระยะขอบ: 40,
ขอบบน: 100,
borderWidth: 1
},
แถว: {
alignItems: 'ศูนย์',
ดิ้น: 1,
flexDirection: 'แถว',
justifyContent: 'ศูนย์กลาง'
},
ข้อความกึ่งกลาง: {
textAlign: 'ศูนย์',
ระยะขอบ: 10,
ขนาดตัวอักษร: 24,
...Platform.select({
ไอโอเอส: {
```

นำเข้าแพลตฟอร์ม

ส่วนประกอบจาก

ตอบสนองพื้นเมือง

Platform.OS ยังสามารถบอกคุณได้

รหัสทำงานบนระบบปฏิบัติการใด

ใช้ประโยชน์จาก .ของคุณ

ความรู้เกี่ยวกับตำแหน่งที่แน่นอน

ใช้ Platform.select เพื่อเลือกสไตล์

สำหรับแพลตฟอร์มที่เหมาะสม

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc

facebook.com/somsacki

หน้า 34


```
fontFamily: 'เครื่องพิมพ์ดีดอเมริกัน'
},
หุ่นยนต์: {
fontFamily: 'monospace'
}
})
}
});
```

ตัวอย่างนี้แสดงวิธีการเลือกแบบอักษรตามระบบปฏิบัติการ แต่ชุดแบบอักษรที่คุณใช้งาน posal ยังคง จำกัด เฉพาะสิ่งที่มาพร้อมกับ React Native นอกกรอบ คุณสามารถเพิ่มกำหนดเอง ฟอนต์ไปยังโปรเจ็กต์โดยใช้ไฟล์ฟอนต์ (TTF, OTF เป็นต้น) และเชื่อมโยงเข้ากับแอพพลิเคชั่นของคุณ เป็นทรัพย์สิน ในทางทฤษฎี กระบวนการนั้นง่าย แต่ความสำเร็จนั้นแตกต่างกันอย่างมากขึ้นอยู่กับ OS และไฟล์ฟอนต์ที่ใช้ อยากให้รู้ว่าทำได้ แต่ถ้าอยากทำ ลองใช้ แยกเครื่องมือค้นหาที่คุณเลือก และค้นหาลิงก์ที่ตอบสนองแบบเนทีฟ

DJUSTING ขนาดตัวอักษรแบบอักษร S IZE

fontSize นั้นค่อนข้างเรียบง่าย: มันปรับขนาดของข้อความในองค์ประกอบ Text คุณเคยใช้ ค่อนข้างมากแล้ว ดังนั้นเราจะไม่ลงรายละเอียดมากไปกว่าข้อเท็จจริงที่ว่า เริ่มต้นFontSizeคือ 14

แบบอักษร C แบบแขวน

คุณสามารถใช้fontStyleเพื่อเปลี่ยนรูปแบบแบบอักษรเป็นตัวเอียง เริ่มต้นเป็น'ปกติ' NS เพียงสองตัวเลือกในขณะนี้'ปกติ'และ'เอียง'

S น้ำหนัก PECIFYING FONT

fontWeight หมายถึงความหนาของฟอนต์ เริ่มต้นเป็น 'ปกติ' หรือ'400' NS ตัวเลือกสำหรับfontWeightคือ'ปกติ' , 'ตัวหนา' , '100' , '200' , '300' , '400' , '500' , '600' , '700' , '800' และ '900 ' ยิ่งค่าน้อย ข้อความยิ่งอ่อน/บางลง ยิ่งค่ามาก ข้อความก็จะยิ่งหนา/เข้มขึ้น

เมื่อคุณรู้วิธีเปลี่ยนรูปแบบแบบอักษรแล้ว คุณก็เกือบจะเสร็จสิ้น Profile ส่วนประกอบของการ์ด มาเปลี่ยนรูปแบบฟอนต์และดูว่าคุณจะเข้าใกล้ . ได้แค่ไหน ผลิตภัณฑ์ขั้นสุดท้ายดังแสดงในรูปที่ 4.17 รายการถัดไปแสดงวิธีการเปลี่ยนสไตล์ จากรายการ 4.16 เพื่อให้ได้รูปลักษณ์นี้

…

ชื่อการ์ด: {
สี: 'ขาว',
fontWeight: 'ตัวหนา',
ขนาดตัวอักษร: 24,
ขอบบน: 30,
},
…
การ์ดอาชีพ: {
fontWeight: 'ตัวหนา',
ขอบบน: 10,
ระยะขอบด้านล่าง: 10,
},
cardDescription: {
เปลี่ยนน้ำหนักแบบอักษร
ของข้อความชื่อเป็นตัวหนา
เปลี่ยนขนาดตัวอักษร
ของข้อความชื่อถึง 24
ตัวหนาข้อความอาชีพ
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

107
การจัดองค์ประกอบข้อความ
fontStyle: 'ตัวเอียง',
ขอบบน: 10,
ขอบขวา: 40,
ระยะขอบซ้าย: 40,
marginBottom: 10
}
…
การปรับเปลี่ยนรูปแบบฟอนต์สำหรับชื่อ อาชีพ และข้อความคำอธิบายช่วย
ดึงดูดแต่ละส่วน แต่ชื่อยังไม่โดดเด่นมากนัก วินาทีถัดมา-
ครอบคลุมถึงวิธีการตกแต่งข้อความและวิธีการใช้เทคนิคเหล่านั้นในการทำ
ชื่อโดดเด่นในบัตรโปรไฟล์

4.3.3 การใช้รูปแบบข้อความตกแต่ง

ในส่วนนี้ คุณจะไปไกลกว่าพื้นฐานของการเปลี่ยนรูปแบบแบบอักษรและเริ่มใช้-
รูปแบบการตกแต่งเป็นข้อความ ฉันจะแสดงวิธีทำสิ่งต่างๆ เช่น ขีดเส้นใต้และขีด-

ผ่านข้อความและเพิ่มเงาตกกระทบ เทคนิคเหล่านี้สามารถเพิ่มความหลากหลายของภาพให้กับ
แอปพลิเคชันและองค์ประกอบข้อความช่วยให้โดดเด่นจากที่อื่น
นี่คือคุณสมบัติที่เราจะกล่าวถึงในส่วนนี้:

- iOS และ Android — lineHeight , textAlign , textDecorationLine , textShadow-Color , textShadowOffset และ textShadowRadius
- Android เท่านั้น — textAlignVertical
- iOS เท่านั้น — letterSpacing , textDecorationColor , textDecorationStyle และ writingDirection

ขอให้สังเกตว่าคุณสมบัติบางอย่างใช้กับระบบปฏิบัติการใดระบบหนึ่งเท่านั้น ค่าบางอย่างที่
สามารถกำหนดคุณสมบัติเฉพาะ OS ได้ สิ่งสำคัญคือต้องเก็บสิ่งนี้ไว้ใน
โดยเฉพาะอย่างยิ่งถ้าคุณใช้รูปแบบเฉพาะเพื่อเน้นองค์ประกอบเฉพาะของ
ข้อความบนหน้าจอ
ทำให้ข้อความอธิบายเป็นตัวเอียง

React Native Developer
John เป็น JavaScript ที่ยอดเยี่ยมจริงๆ
นักพัฒนา เขาชอบใช้ JS to
สร้างแอปพลิเคชัน React Native
สำหรับ iOS และ Android
รูปที่ 4.17 การ์ดโปรไฟล์ที่มีรูปแบบแบบอักษรที่ใช้กับ
ข้อความชื่อ อาชีพ และคำอธิบาย

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 36

S PECIFYING ความสูงขององค์ประกอบข้อความ

lineHeight ระบุความสูงขององค์ประกอบข้อความ รูปที่ 4.18 และรายการ 4.18 แสดง
ตัวอย่างลักษณะการทำงานที่แตกต่างกันบน iOS กับ Android lineHeightของ
100 ใช้กับองค์ประกอบข้อความ B: ความสูงของบรรทัดนั้นมากกว่า
คนอื่น ๆ. สังเกตด้วยว่า iOS และ Android วางตำแหน่งข้อความภายในบรรทัดต่างกันอย่างไร-
เอนทิตี ใน Android ข้อความจะอยู่ที่ด้านล่างของบรรทัด

นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { Platform, StyleSheet, Text, View} จาก 'react-native';
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {

```
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<TextContainer>
<LeftText>ข้อความ A</LeftText>
</TextContainer>
<TextContainer>
<รูปแบบข้อความด้านซ้าย={{lineHeight: 100}}>
ข้อความ B
</LeftText>
</TextContainer>
<TextContainer>
<LeftText>ข้อความ C</LeftText>
</TextContainer>
<TextContainer>
<LeftText>{Platform.OS}</LeftText>
</TextContainer>
</ดู>
);
}
}
const LeftText = (อุปกรณ์ประกอบฉาก) => (
<Text style={[styles.leftText, props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</Text>
);
const TextContainer = (อุปกรณ์ประกอบฉาก) => (
<ดู style={[styles.textContainer, props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
```

รูปที่ 4.18 ตัวอย่างการใช้ lineHeight บน iOS และแอนดรอยด์

ตั้งค่า lineHeight เป็น 100

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc facebook.com/somsacki

---

หน้า 37


```
</ดู>
);
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ความกว้าง: 300,
```

ความสูง: 300,
ระยะขอบ: 40,
ขอบด้านบน: 100
},
คอนเทนเนอร์ข้อความ: {
borderWidth: 1
},
ข้อความซ้าย: {
ขนาดตัวอักษร: 20
}
});

ข้อความเรียงตามแนวนอน

textAlign หมายถึงข้อความในองค์ประกอบจะถูกจัดแนวตามแนวนอน ตัวเลือก
สำหรับtextAlignคือ'auto' , 'center' , 'right' , 'left'และ'justify' ('justify'คือ
iOS เท่านั้น)

U การขีดเส้นใต้ข้อความหรือการเพิ่มบรรทัดผ่านข้อความ

ใช้คุณสมบัติtextDecorationLineเพื่อเพิ่มขีดเส้นใต้หรือเส้นผ่าน
ข้อความที่กำหนด ตัวเลือกสำหรับtextDecorationLineคือ'none' , 'underline' , 'line-
ผ่าน' และ'ขีดเส้นใต้บรรทัดผ่าน' ค่าเริ่มต้นคือ'ไม่มี' เมื่อคุณ
ระบุ'ขีดเส้นใต้บรรทัดผ่าน'ช่องว่างเดียวแยกค่าในเครื่องหมายคำพูด

T EXT - รูปแบบการตกแต่ง (I OS เท่านั้น)

iOS รองรับสไตล์การตกแต่งข้อความหลายแบบที่ Android ไม่รองรับ อย่างแรกคือข้อความ-
DecorationColor ซึ่งช่วยให้คุณสามารถตั้งค่าสีสำหรับtextDecorationLine iOS ด้วย
รองรับการจัดแต่งแนวเส้นนั่นเอง บน Android บรรทัดอยู่เสมอของแข็ง แต่บน iOS หัวจักร
tDecorationStyle ช่วยให้คุณระบุ 'แข็ง' , 'คู่' , 'จุด' และ'ประ
Android จะไม่สนใจสไตล์เพิ่มเติมเหล่านี้
หากต้องการใช้รูปแบบการตกแต่ง iOS เพิ่มเติม ให้ระบุร่วมกับ
สไตล์textDecorationLineหลัก ตัวอย่างเช่น:

{
textDecorationLine: 'ขีดเส้นใต้',
textDecorationColor: 'สีแดง',
textDecorationStyle: 'สองเท่า'
}

DDING SHADOWS กับข้อความ

คุณสามารถใช้textShadowColor , textShadowOffsetและtextShadowRadius prop-
erties เพื่อเพิ่มเงาให้กับองค์ประกอบข้อความ ในการสร้างเงา คุณต้องระบุ
สามสิ่ง:

- สี
- ออฟเซ็ต
- รัศมี

การตั้งค่าเส้นขอบช่วยให้คุณได้อย่างง่ายดาย

ดูความสูงของเส้น

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 38

ออฟเซ็ตระบุตำแหน่งของเงาที่สัมพันธ์กับส่วนประกอบที่หล่อ
เงา. รัศมีโดยทั่วไปกำหนดว่าเงาปรากฏอย่างไร คุณสามารถระบุ a
เงาข้อความเช่นนี้:

```
{
textShadowColor: 'สีแดง',
textShadowOffset: {ความกว้าง: -2, ความสูง: -2},
textShadowRadius: 4
}
```

C การเว้นวรรคจดหมายควบคุม (I OS เท่านั้น)

letterSpacing ระบุระยะห่างระหว่างอักขระข้อความ ไม่ใช่สิ่งที่คุณจะ
ใช้ทุกวัน แต่สามารถสร้างเอฟเฟกต์ภาพที่น่าสนใจได้ จำไว้ว่ามันคือ
iOS เท่านั้น ดังนั้นใช้หากคุณต้องการ

E ตัวอย่างรูปแบบข้อความ

เราได้ผ่านรูปแบบต่างๆ มากมายในส่วนนี้ รูปที่ 4.19 แสดงต่างๆ
สไตล์ที่ใช้กับองค์ประกอบข้อความ
ต่อไปนี้คือรูปแบบคร่าวๆ ที่ใช้สำหรับแต่ละตัวอย่างในภาพที่ 4.19:

- เป็นตัวเอียงใช้{fontStyle: 'เอียง'}
- B แสดงการตกแต่งข้อความด้วยการขีดเส้นใต้และเส้นผ่านข้อความ รูปแบบ
สำหรับเรื่องนี้คือ{textDecorationLine: 'ขีดเส้นใต้บรรทัดผ่าน'}
- C ขยายจากตัวอย่าง B โดยใช้รูปแบบข้อความสำหรับ iOS เท่านั้น{textDec-
orationColor: 'red', textDecorationStyle: 'จุด'} สังเกตว่าสิ่งเหล่านี้
สไตล์ไม่มีผลใน Android
- D ใช้เงาโดยใช้{textShadowColor: 'red', textShadowOffset:

{ความกว้าง: -2, ความสูง: -2} textShadowRadius: 4}

- E ใช้{letterSpacing: 5} สำหรับiOS เท่านั้นซึ่งไม่มีผลกับ Android
- ข้อความ iOS และ Android ถูกจัดรูปแบบโดยใช้{textAlign: 'center', fontWeight: 'ตัวหนา'} .

ใช้รายการ 4.19 เป็นจุดเริ่มต้น และดูว่าการปรับเปลี่ยนสไตล์ส่งผลต่อผลลัพธ์อย่างไร

รูปที่ 4.19 ตัวอย่างต่างๆ ของส่วนประกอบข้อความการจัดรูปแบบ

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 39

111

การจัดองค์ประกอบข้อความ

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { Platform, StyleSheet, Text, View} จาก 'react-native';
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<LeftText style={{fontStyle: 'italic'}}>
ก) ตัวเอียง
</LeftText>
<LeftText style={{textDecorationLine: 'underline line-through'}}>
B) ขีดเส้นใต้และขีดเส้นผ่าน
</LeftText>
<LeftText style={{textDecorationLine: 'ขีดเส้นใต้เส้นผ่าน',
textDecorationColor: 'สีแดง',
textDecorationStyle: 'dotted'}}>
C) ขีดเส้นใต้และขีดเส้นผ่าน
</LeftText>
<รูปแบบข้อความด้านซ้าย={{textShadowColor: 'สีแดง',
textShadowOffset: {ความกว้าง: -2, ความสูง: -2},
textShadowRadius: 4}}>
D) ข้อความเงา
</LeftText>
<รูปแบบข้อความด้านซ้าย={{letterSpacing: 5}}>
E) การเว้นวรรคตัวอักษร
</LeftText>
<รูปแบบข้อความด้านซ้าย={{textAlign: 'center', fontWeight: 'bold'}}>
{Platform.OS}
</LeftText>
```

```
</ดู>
);
}
}
const LeftText = (อุปกรณ์ประกอบฉาก) => (
<Text style={[styles.leftText, props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</Text>
);
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ความกว้าง: 300,
ความสูง: 300,
ระยะขอบ: 40,
ขอบด้านบน: 100
},
ข้อความซ้าย: {
ขนาดตัวอักษร: 20,
paddingBottom: 10
}
});
```

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 40

ตอนนี้คุณรู้วิธีสร้างเอฟเฟกต์เงาแล้ว มาเพิ่มเงาให้กับบุคคลกันเถอะ
ชื่อจึงโดดเด่นกว่าข้อความอื่น รูปที่ 4.20 แสดงผลที่ต้องการ
รหัสที่สมบูรณ์สำหรับบัตรโปรไฟล์จะมีให้ในครั้งต่อไป คุณต้องเพิ่ม . เท่านั้น
ตัวอย่างเล็ก ๆ เพื่อกำหนดเงาข้อความสำหรับชื่อ

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { รูปภาพ, สไตล์ชีต, ข้อความ, มุมมอง} จาก 'react-native';
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<ดู style={styles.cardContainer}>
<ดู style={styles.cardImageContainer}>
<รูปแบบรูปภาพ={styles.cardImage}
```

source={require('./user.png')}/>
</ดู>
<ดู>
<Text style={styles.cardName}>
จอห์น โด
</Text>
</ดู>
<ดู style={styles.cardOccupationContainer}>
<Text style={styles.cardOccupation}>
React Native Developer
</Text>
</ดู>
<ดู>
<Text style={styles.cardDescription}>
John เป็นนักพัฒนา JavaScript ที่ยอดเยี่ยมจริงๆ
เขาชอบใช้ JS เพื่อสร้าง React Native
แอปพลิเคชันสำหรับ iOS และ Android

React Native Developer
John เป็น JavaScript ที่ยอดเยี่ยมจริงๆ
นักพัฒนา เขาชอบใช้ JS to
สร้างแอปพลิเคชัน React Native
สำหรับ iOS และ Android
รูปที่ 4.20 ตัวอย่าง Profile Card ที่กรอกเสร็จแล้ว ข้อความ
มีการเพิ่มข้อมูลเกี่ยวกับบุคคลที่ใช้ข้อความ
เทคนิคการจัดแต่งทรงผมที่ครอบคลุมในส่วนนี้

หน้า 41

</Text>
</ดู>
</ดู>
</ดู>
);
}
}
const profileCardColor = 'dodgerblue';
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ดิ้น: 1,

```
justifyContent: 'ศูนย์',
alignItems: 'ศูนย์'
},
คอนเทนเนอร์การ์ด: {
alignItems: 'ศูนย์',
borderColor: 'สีดำ',
ขอบกว้าง: 3,
borderStyle: 'ของแข็ง',
รัศมีชายแดน: 20,
พื้นหลังสี: profileCardColor,
ความกว้าง: 300,
ส่วนสูง: 400
},
cardImageContainer: {
alignItems: 'ศูนย์',
พื้นหลังสี: 'สีขาว',
ขอบกว้าง: 3,
borderColor: 'สีดำ',
ความกว้าง: 120,
ความสูง: 120,
รัศมีขอบ: 60,
ขอบบน: 30,
paddingTop: 15
},
ภาพการ์ด: {
ความกว้าง: 80,
ส่วนสูง: 80
},
ชื่อการ์ด: {
สี: 'ขาว',
fontWeight: 'ตัวหนา',
ขนาดตัวอักษร: 24,
ขอบบน: 30,
textShadowColor: 'ดำ',
textShadowOffset: {
ความสูง: 2,
ความกว้าง: 2
},
textShadowRadius: 3
},
การ์ดอาชีพคอนเทนเนอร์: {
borderColor: 'สีดำ',
borderBottomWidth: 3
ตั้งค่าสีเงาเป็นสีดำ
```

บนองค์ประกอบข้อความชื่อเรื่อง
ตั้งค่าชดเชยเงาเป็น
ลงและไปทางขวา
กำหนดรัศมีเงา
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

```
},
การ์ดอาชีพ: {
fontWeight: 'ตัวหนา',
ขอบบน: 10,
ระยะขอบด้านล่าง: 10,
},
cardDescription: {
fontStyle: 'ตัวเอียง',
ขอบบน: 10,
ขอบขวา: 40,
ระยะขอบซ้าย: 40,
marginBottom: 10
}
});
```

มีหลายอย่างที่คุณสามารถทำได้กับตัวอย่างพื้นฐานนี้เพื่อให้ดียิ่งขึ้นไปอีก แต่เป้าหมายคือเพื่อแสดงให้เห็นว่าการเข้าใจแนวคิดการจัดสไตล์มีประโยชน์เพียงใด ไม่ต้องเป็นแฟนนักออกแบบกราฟิกที่มีไหวพริบเพื่อสร้างองค์ประกอบที่ดูดี—เทคนิคง่ายๆ สองสามข้อสามารถทำให้ใบสมัครของคุณดูดีได้

เราครอบคลุมเนื้อหามากมายในบทนี้ แต่เชื่อหรือไม่ นี่เป็นเรื่องสั้น

การแนะนำ! เราจะสำรวจหัวข้อขั้นสูงเพิ่มเติมในบทที่ 5

สรุป

- สามารถใช้สไตล์แบบอินไลน์กับส่วนประกอบหรือโดยการสร้างสไตล์ชีตที่สามารถถูกอ้างอิงโดยส่วนประกอบ
- รูปแบบควรจัดอยู่ในไฟล์เดียวกับองค์ประกอบหลังองค์ประกอบ

nent definition หรือภายนอกเป็นไฟล์ styles.js แยกต่างหาก

- ลักษณะเป็นรหัส ความจริงที่ว่า JavaScript เป็นภาษาที่สมบูรณ์พร้อมตัวแปรและฟังก์ชันมีประโยชน์มากมายเหนือ CSS แบบเดิม

▪ ดูส่วนประกอบเป็นส่วนประกอบหลักของ UI และมีหลายสไตล์-
ไอเอ็นจีคุณสมบัติ
▪ คุณสามารถใช้เส้นขอบได้หลายวิธีเพื่อปรับปรุงรูปลักษณ์ของส่วนประกอบ คุณสามารถ
แม้กระทั่งใช้เส้นขอบเพื่อสร้างรูปร่าง เช่น วงกลม
▪ คุณสามารถใช้ระยะขอบและช่องว่างภายในเพื่อจัดตำแหน่งส่วนประกอบที่สัมพันธ์กัน
▪ การวางตำแหน่งแบบสัมบูรณ์ช่วยให้คุณวางส่วนประกอบไว้ที่ใดก็ได้ภายในพาเรนต์
คอนเทนเนอร์.
▪ คลิปสามารถเกิดขึ้นได้บนอุปกรณ์ Android ขึ้นอยู่กับว่าคุณตั้งค่าเส้นขอบ
gins และช่องว่างภายใน
▪ การระบุแบบอักษรอื่นที่ไม่ใช่ค่าเริ่มต้นอาจเป็นเรื่องยุ่งยาก ใช้ส่วนประกอบแพลตฟอร์ม-
nent เพื่อเลือกแบบอักษรที่เหมาะสมสำหรับระบบปฏิบัติการ
▪ ใช้รูปแบบตัวอักษรทั่วไป เช่น สี ขนาด และน้ำหนัก เพื่อเปลี่ยนขนาดและลักษณะที่ปรากฏ-
ance ขององค์ประกอบข้อความ
▪ มีการเรนเดอร์ความแตกต่างระหว่าง OSs เช่นความสูงของบรรทัดอย่างไร
ทำงานแตกต่างกันระหว่าง iOS และ Android
▪ รูปแบบการตกแต่งข้อความสามารถเพิ่มขีดเส้นใต้หรือวางเงาให้กับข้อความได้ ชุดของ
สไตล์ที่ใช้ได้จะแตกต่างกันไปในแต่ละระบบปฏิบัติการ
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

115

จัดแต่งทรงผมอย่างล้ำลึก

บทนี้ครอบคลุม

- ขนาดและรูปแบบเฉพาะแพลตฟอร์ม
- การเพิ่มเงาให้กับส่วนประกอบ
- เคลื่อนย้ายและหมุนส่วนประกอบบน

x8 และ y8axes

- ส่วนประกอบมาตราส่วนและส่วนเอียง
- การใช้ exbox สำหรับเลย์เอาต์

บทที่ 4 แนะนำการจัดสไตล์ส่วนประกอบ React Native แสดงวิธีการจัดสไตล์View
และองค์ประกอบข้อความสไตล์ที่คุณน่าจะใช้ทุกวันและที่ส่วนใหญ่ส่งผลต่อ
รูปลักษณ์ขององค์ประกอบ บทนี้ยังคงอภิปรายและเข้าสู่มากขึ้น
ความลึกด้วยรูปแบบเฉพาะแพลตฟอร์ม เงาตก; จัดการส่วนประกอบด้วย
การแปลง เช่น การแปล การหมุน การปรับขนาด และการเอียง และแบบไดนามิก
การจัดวางส่วนประกอบด้วย flexbox
บางหัวข้อเหล่านี้อาจรู้สึกคุ้นเคย คุณใช้รูปแบบเฉพาะแพลตฟอร์มและความยืดหยุ่น
กล่องในหลายตัวอย่างในบทที่ 4 เราไม่ได้กล่าวถึงในรายละเอียด แต่คุณ
เห็นพวกเขาในรายการรหัสสองสามรายการ
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 44

บทนี้จะขยายในหัวข้อเหล่านั้น การแปลงร่างให้อำนาจแก่คุณในการจัดการ-
เลือส่วนประกอบในสองหรือสามมิติ คุณสามารถแปลส่วนประกอบจากหนึ่ง
วางตำแหน่งอื่น หมุนส่วนประกอบ ปรับขนาดส่วนประกอบให้มีขนาดต่างกัน และเอียง
ส่วนประกอบ การแปลงร่างมีประโยชน์ในตัวเอง แต่พวกมันจะเล่นได้ยิ่งใหญ่กว่ามาก
บทบาทในบทที่ 7 ซึ่งกล่าวถึงอนิเมชั่นโดยละเอียด
เราจะพูดถึงความแตกต่างระหว่างแพลตฟอร์มและรูปลักษณ์ต่อไป
ลึกยิ่งขึ้นที่ flexbox เนื่องจาก flexbox เป็นแนวคิดพื้นฐาน จึงเป็นสิ่งสำคัญที่จะ
ทำความเข้าใจอย่างถูกต้องเพื่อให้คุณสามารถสร้างเลย์เอาต์และ UI ใน React Native คุณจะต้องประห-
ใช้ flexbox ได้ในทุกแอปพลิเคชันที่คุณสร้าง คุณจะใช้สไตล์ใหม่ของคุณ
เทคนิคในการสร้างคุณสมบัติใหม่ต่อไปในตัวอย่างProfileCardจาก
บทที่แล้ว

5.1 ขนาดและรูปแบบเฉพาะแพลตฟอร์ม

คุณได้เห็นวิธีการใช้ฟังก์ชั่นยูทิลิตี้Platform.selectเพื่อเลือกแบบอักษรที่พร้อมใช้งาน
เฉพาะบน iOS หรือ Android คุณใช้Platform.selectเพื่อเลือกแบบอักษรแบบโมโนสเปซ
สนับสนุนโดยแต่ละแพลตฟอร์ม คุณอาจไม่ได้คิดอะไรมากในขณะนั้น แต่
สิ่งสำคัญคือต้องจำไว้ว่าคุณกำลังพัฒนาสำหรับสองแพลตฟอร์มที่แตกต่างกัน NS
สไตล์ที่คุณใช้กับส่วนประกอบอาจมีลักษณะหรือทำงานแตกต่างกันระหว่างสอง OS หรือ
แม้กระทั่งระหว่าง iOS และ Android เวอร์ชันต่างๆ
คุณไม่ได้เข้ารหัสสำหรับอุปกรณ์เครื่องเดียว คุณไม่ได้เขียนโค้ดสำหรับระบบปฏิบัติการเดียว NS
ความงามของ React Native คือคุณกำลังใช้ JavaScript เพื่อสร้างแอปพลิเคชันที่สามารถทำงานได้
ทั้งบน iOS และ Android หากคุณดูเอกสาร React Native คุณจะ
ดูส่วนประกอบต่างๆ ที่ต่อท้ายด้วย IOS หรือ Android เช่นProgressBarAndroid ,
ProgressViewIOS และ ToolbarAndroid จึงไม่น่าแปลกใจเลยที่สไตล์จะสามารถทำได้
เป็นแพลตฟอร์มเฉพาะด้วย
คุณอาจไม่ได้สังเกตว่าคุณไม่เคยระบุขนาดเป็นพิกเซลสำหรับสิ่งใด
เช่นความกว้าง: 300เทียบกับความกว้าง: 300px ' นั่นก็เพราะว่าแนวคิดเรื่องขนาดก็ต่างกัน
ระหว่างระบบปฏิบัติการ iOS และ Android

5.1.1 พิกเซล จุด และ DPs

ขนาดอาจเป็นหัวข้อที่สับสน แต่สิ่งสำคัญคือต้องจำไว้ถ้าคุณต้องการ
แม่นยำอย่างยิ่งเมื่อวางตำแหน่งส่วนประกอบบนหน้าจอ แม้ว่าคุณจะไม่ใช่
พยายามสร้างเลย์เอาต์ที่มีความเที่ยงตรงสูงจะเป็นประโยชน์ในการทำความเข้าใจแนวคิดใน
กรณีที่คุณพบความคลาดเคลื่อนเล็กน้อยในเค้าโครงของคุณจากอุปกรณ์เครื่องหนึ่งไปยังอีกเครื่องหนึ่ง
เริ่มจากจุดเริ่มต้นและกำหนดพิกเซล พิกเซลเป็นหน่วยที่เล็กที่สุดของโปรแกรม-
สีตารางบนจอแสดงผล พิกเซลมักประกอบด้วยสีแดง สีเขียว และสีน้ำเงิน (RGB)
ส่วนประกอบ โดยการจัดการความเข้มของค่า RGB แต่ละค่า พิกเซลจะปล่อยสีที่คุณ
ดู. พิกเซลไม่ได้บอกอะไรคุณจนกว่าคุณจะเริ่มดูคุณสมบัติทางกายภาพของ
จอแสดงผล: ขนาดหน้าจอ ความละเอียด และจุดต่อนิ้ว
ขนาดหน้าจอคือการวัดในแนวทแยงของหน้าจอจากมุมหนึ่งไปอีกมุมหนึ่ง
ตัวอย่างเช่น ขนาดหน้าจอดั้งเดิมของ iPhone คือ 3.5 นิ้ว ในขณะที่ขนาดหน้าจอของ
iPhone X มีขนาด 5.8 นิ้ว แม้ว่า iPhone X จะใหญ่กว่ามาก แต่ขนาดไม่ได้
มีความหมายอะไรจนกว่าคุณจะเข้าใจจำนวนพิกเซลที่พอดีกับขนาดหน้าจอนั้น

117

ขนาดและรูปแบบเฉพาะแพลตฟอร์ม

ความละเอียดคือจำนวนพิกเซลในจอแสดงผล ซึ่งโดยทั่วไปจะแสดงเป็น
จำนวนพิกเซลตามความกว้างและความสูงของอุปกรณ์ iPhone เดิมคือ
320 × 480 ในขณะที่ iPhone X คือ 1125 × 2436
ขนาดหน้าจอและความละเอียดสามารถใช้ในการคำนวณความหนาแน่นของพิกเซล: พิกเซลต่อ
นิ้ว (PPI) คุณมักจะเห็นสิ่งนี้แสดงเป็นจุดต่อนิ้ว (DPI) ซึ่งเป็นคำที่ถือไว้
จากโลกแห่งการพิมพ์ซึ่งจุดสีถูกพิมพ์ลงบนหน้ากระดาษ PPI และ DPI คือ
มักใช้สลับกันได้แม้จะไม่ถูกต้องนัก ดังนั้นหากคุณเห็นDPI
ใช้ในการอ้างอิงถึงหน้าจอ รู้ว่า PPI คือสิ่งที่กำลังถูกกล่าวถึงอย่างแท้จริง
PPI เป็นตัววัดความคมชัดของภาพ ลองนึกภาพถ้าสองหน้าจอมีเหมือนกัน
ความละเอียด 320 × 480 (ครึ่ง VGA) ภาพเดียวกันจะมีลักษณะอย่างไรในขนาด 3.5 นิ้ว
จอแสดงผล iPhone กับจอภาพ HVGA ขนาด 17 นิ้ว? ภาพเดียวกันจะดูมาก
คมชัดกว่าบน iPhone เพราะมี 163 PPI เทียบกับจอภาพ CRT ซึ่งมี 34 PPI
คุณสามารถใส่ข้อมูลได้เกือบห้าเท่าในพื้นที่ทางกายภาพเดียวกันกับต้นฉบับ
ไอโฟน. ตารางที่ 5.1 เปรียบเทียบขนาดแนวทแยง ความละเอียด และ PPI ของอุปกรณ์ทั้งสอง
ตารางที่ 5.1 การเปรียบเทียบ PPI ของจอภาพ HVGA ขนาด 17 นิ้ว กับ PPI ของ iPhone รุ่นดั้งเดิม

17 นิ้ว
3.5 นิ้ว

320 480
320 480

34
163

ทำไมเรื่องนี้? เพราะทั้ง iOS และ Android ไม่ได้ใช้การวัดทางกายภาพที่แท้จริง-
ความมั่นใจในการแสดงเนื้อหาไปยังหน้าจอของอุปกรณ์ iOS ใช้การวัดนามธรรมของ
และ Android ใช้การวัดพิกเซลที่ไม่ขึ้นกับความหนาแน่นแบบนามธรรมที่คล้ายกัน
เมื่อ iPhone 4 มาถึงที่เกิดเหตุ มันมีขนาดเท่ากับรุ่นก่อน-
ซอ; แต่มีหน้าจอ Retina ใหม่ที่สวยงามด้วยความละเอียด 640 × 960 สี่เท่า
ความละเอียดของอุปกรณ์เดิม หาก iPhone ได้แสดงภาพจากที่มีอยู่
แอพในอัตราส่วน 1:1 ทุกอย่างจะถูกวาดในขนาดหนึ่งในสี่ของจอภาพ Retina ใหม่
เล่น. มันคงเป็นเรื่องที่บ้ามากสำหรับ Apple ที่จะทำการเปลี่ยนแปลงดังกล่าวและ

ทำลายแอพที่มีอยู่ทั้งหมด
แต่ Apple ได้แนะนำแนวคิดเชิงตรรกะของประเด็น จุดคือหน่วยของระยะทาง
ที่สามารถปรับขนาดได้อย่างอิสระตามความละเอียดของอุปกรณ์ ดังนั้นภาพขนาด 320 × 480 ที่ถ่าย
ขยายขนาดหน้าจอทั้งหมดบน iPhone เดิมได้ถึง 2 เท่าเพื่อให้พอดีกับ Ret-
ในการแสดงผล รูปที่ 5.1 แสดงภาพความหนาแน่นของพิกเซลสำหรับ iPhone หลายรุ่น
163 PPI ดั้งเดิมของ iPhone เป็นพื้นฐานสำหรับจุด iOS จุด iOS คือ 1/163
นิ้ว โดยไม่ต้องลงรายละเอียดเพิ่มเติม Android ใช้มาตรการที่คล้ายกันที่เรียกว่าa
พิกเซลที่ไม่ขึ้นกับอุปกรณ์ (DIP มักย่อว่า DP) Android DP คือ 1/160 นิ้ว
เมื่อกำหนดสไตล์ใน React Native คุณจะใช้แนวคิดเชิงตรรกะของพิกเซล จุด
บน iOS และ DP บน Android เมื่อทำงานในระดับท้องถิ่น คุณอาจ
ต้องทำงานกับพิกเซลของอุปกรณ์โดยการคูณพิกเซลตรรกะด้วยมาตราส่วนหน้าจอ (สำหรับ
ตัวอย่าง 2x, 3x)
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 46

5.1.2 การสร้างเงาตกกระทบด้วย ShadowPropTypesIOS และ Elevation

ในบทที่ 4 คุณใช้คุณสมบัติเงาข้อความเพื่อเพิ่มเงาให้กับ
ชื่อโปรไฟล์การ์ด ทั้ง iOS และ Android รองรับการเพิ่มเงาให้กับ Text
ส่วนประกอบ. คงจะดีถ้าจะเพิ่มProfileCard ให้มากขึ้นด้วยการเพิ่ม drop
เงาบนการ์ดและคอนเทนเนอร์รูปทรงกลม แต่ไม่มีรูปแบบทั่วไป
คุณสมบัติสำหรับส่วนประกอบViewเพื่อใช้ระหว่างสองแพลตฟอร์ม
ไม่ได้หมายความว่าทั้งหมดจะหายไป ShadowPropTypesIOSสไตล์สามารถนำมาใช้เพื่อเพิ่ม
เงาตกกระทบบนอุปกรณ์ iOS; ไม่ส่งผลต่อลำดับ z ของส่วนประกอบ บน
Android คุณสามารถใช้รูปแบบElevationเพื่อจำลองเงาตกกระทบ แต่มันมีผล
z-order ของส่วนประกอบ

C REATING DROP SHADOWS ใน I OS ด้วย S HADOW P ROP T YPES IOS

มาดูวิธีใช้สไตล์ShadowPropTypesIOSเพื่อเพิ่มเงาตกหล่นในมุมมองบางส่วนกัน
ส่วนประกอบ รูปที่ 5.2 แสดงเอฟเฟกต์เงาต่างๆ ที่สามารถทำได้ ตาราง 5.2
แสดงรายการการตั้งค่าเฉพาะที่ใช้เพื่อให้ได้เอฟเฟกต์เงาแต่ละอัน ประเด็นสำคัญ
มีรายละเอียดดังนี้:

- หากคุณไม่ระบุค่าสำหรับshadowOpacity คุณจะไม่เห็นเงา
- การชดเชยเงาจะแสดงเป็นความกว้างและความสูง แต่คุณสามารถนึกถึง

เป็นการเคลื่อนเงาไปในทิศทาง x และ y คุณยังสามารถระบุค่าลบ

ค่าความกว้างและความสูง

- shadowOpacity 1 เป็นของแข็งอย่างสมบูรณ์ในขณะที่มูลค่า 0.2 มีมากขึ้น

โปร่งใส.

- ค่าของshadowRadius จะเบลอขอบของเงาได้อย่างมีประสิทธิภาพ เงา

จะกระจายตัวมากขึ้น

1x

=

1 POINT = 1 พิกเซล

ไอโฟนแท้

2x

=

1 POINT = 4 พิกเซล

iPhone 4,5,6,7,8

3x

=

1 POINT = 9 พิกเซล

iPhone 6+,7+,8+,X

รูปที่ 5.1 ภาพแสดงจุดเปรียบเทียบกับความหนาแน่นของพิกเซลสำหรับ iPhone NS

iPhone ดั้งเดิมมีความละเอียด 320 × 480 iPhone 4 มีความละเอียด

640 × 960, ความละเอียดสี่เท่าของอุปกรณ์ดั้งเดิม iPhone 4 มีสองครั้ง

PPI (326 เทียบกับ 163) ดังนั้นรูปภาพจึงถูกปรับขนาดขึ้น 2 เท่า

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc

facebook.com/somsacki

หน้า 47

119

ขนาดและรูปแบบเฉพาะแพลตฟอร์ม

ตารางที่ 5.2 คุณสมบัติเงาที่ใช้สร้างตัวอย่างในรูปที่5.2

1

สีดำ

10

10

2

สีดำ
10
10
1
3
สีดำ
20
20
1
4
สีดำ
20
20
1
20
5
สีดำ
20
20
0.2
6
สีแดง
20
20
1
7
สีดำ
20
8
สีดำ
85
85
1
รหัสสำหรับรูปนี้สามารถพบได้ในที่เก็บ git ภายใต้บทที่ 5 / ตัวเลข /
รูปที่ 5.2-ShadowPropTypesIOS หากคุณรันโค้ดสำหรับตัวอย่างนี้ อย่าลืม
เรียกใช้ในโปรแกรมจำลอง iOS บนอุปกรณ์ Android คุณจะเห็นสี่เหลี่ยมที่น่าเบื่อแปดช่อง
ด้วยมุมโค้งมน สไตล์ShadowPropTypesIOSจะถูกละเว้นบน Android
PPROXIMATING เงาบนNDROID DEVICES ด้วยความสูง
คุณจะได้รับผลกระทบเช่นเดียวกันกับอุปกรณ์ Android อย่างไร ความจริงก็คือคุณไม่สามารถ คุณสามารถ
ใช้รูปแบบความสูงของ Android เพื่อส่งผลต่อลำดับ z ของส่วนประกอบ ถ้าสองคนขึ้นไป
ตัวอย่าง 1
ตัวอย่างที่ 5
ตัวอย่างที่ 6
ตัวอย่าง 7
ตัวอย่างที่ 8

ตัวอย่าง 2
ตัวอย่างที่ 3
ตัวอย่างที่ 4
รูปที่ 5.2 ตัวอย่างเฉพาะ iOS ของวิธีการใช้สไตล์ ShadowPropTypesIOS กับ View
ส่วนประกอบ ตัวอย่างที่ 1 ใช้เงา แต่ไม่มีการตั้งค่าความทึบซึ่งทำให้เกิดการตก
เงาที่จะไม่ปรากฏ ตัวอย่างที่ 2 มีเอฟเฟกต์เงาเหมือนกัน แต่ตั้งค่าความทึบเป็น 1
ตัวอย่างที่ 3 มีเงาที่ใหญ่กว่าเล็กน้อย และตัวอย่างที่ 4 มีเงาขนาดเดียวกันกับ a
รัศมีเงา ตัวอย่างที่ 5 มีขนาดเงาเท่ากัน แต่ความทึบเปลี่ยนจาก 1 เป็น 0.2
ตัวอย่างที่ 6 เปลี่ยนสีของเงา ตัวอย่างที่ 7 แสดงเงาที่ใช้เท่านั้น
ทิศทางเดียว และตัวอย่างที่ 8 แสดงเงาที่ใช้ไปในทิศทางตรงกันข้าม

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 48

120
C HAPTER 5 จัดแต่งทรงผมอย่างล้ำลึก
ส่วนประกอบใช้พื้นที่เดียวกัน คุณสามารถตัดสินใจได้ว่าอันไหนควรอยู่ข้างหน้าโดย
ให้ระดับความสูงที่ใหญ่กว่าและดังนั้นดัชนี z ที่ใหญ่กว่าซึ่งจะสร้างขนาดเล็ก
เงาตกกระทบ แต่ไม่โดดเด่นเท่าเอฟเฟกต์เงาที่คุณสามารถทำได้บน iOS
โปรดทราบว่าสิ่งนี้ใช้ได้กับ Android เท่านั้น เนื่องจาก iOS ไม่รองรับการยกระดับ
สไตล์และยินดีจะละเว้นหากมีการระบุไว้
อย่างไรก็ตาม มาดูการดำเนินการยกระดับกัน ในการทำเช่นนั้น คุณจะต้องสร้างView compo-
กระเต็นสามกล่องซึ่งแต่ละกล่องวางอย่างเรียบร้อย คุณจะให้พวกเขาสาม
ระดับความสูงที่แตกต่างกัน—1, 2, และ 3—จากนั้น คุณจะย้อนกลับการมอบหมายของลิฟต์-
และดูว่ามีผลกับการจัดวางอย่างไร รูปที่ 5.3 แสดงผลการยกระดับเหล่านี้
การปรับ
ตารางที่ 5.3 แสดงตำแหน่งสัมบูรณ์และระดับความสูงที่ใช้สำหรับกล่องแต่ละกลุ่ม
สังเกตว่าไม่มีอะไรเปลี่ยนแปลงยกเว้นระดับความสูงที่กำหนดให้กับแต่ละช่อง
iOS ละเว้นสไตล์และแสดงกล่อง C ที่ด้านบนของกล่อง B และกล่อง B ที่ด้านบนของกล่อง
A. แต่ Android เคารพสไตล์และพลิกลำดับการแสดงกล่อง ดังนั้น
กล่อง A อยู่ด้านบนของกล่อง B และกล่อง B อยู่ด้านบนของกล่อง C
ตาราง 5.3 การตั้งค่าระดับความสูงสำหรับรูปที่ 5.3

NS
สีแดง
0
0
1
NS
ส้ม
20
2
ค
สีฟ้า
40
40
3
NS
สีแดง
0
0
3
NS
ส้ม
20
20
2
ค
สีฟ้า
40
4

รูปที่ 5.3 ตัวอย่างการใช้รูปแบบการยกระดับบน iOS และ Android บน iOS ระดับความสูงจะถูกละเว้น ส่วนประกอบทั้งหมดยังคงลำดับ z เดียวกัน ดังนั้นองค์ประกอบใดก็ตามที่อยู่ในเลย์เอาต์สุดท้ายจะอยู่ด้านบน บน Android ใช้ระดับความสูงและลำดับ z เปลี่ยนไป ในตัวอย่างที่สอง ที่ระดับความสูง การมอบหมายถูกย้อนกลับ A อยู่ด้านบน

121

ขนาดและรูปแบบเฉพาะแพลตฟอร์ม

5.1.3 นำไปปฏิบัติ: ปล่อยเงาใน ProfileCard

กลับไปที่ตัวอย่างProfileCardจากบทที่แล้วและเพิ่ม drop shad-
ที่จะดูดีบน iOS และไม่ค่อยดีนักบน Android คุณจะเพิ่มเงาให้
คอนเทนเนอร์ProfileCardทั้งหมดและคอนเทนเนอร์รูปภาพแบบวงกลม รูปที่ 5.4 แสดง
สิ่งที่คุณกำลังถ่ายทำบน iOS และสิ่งที่คุณจะได้รับบน Android
สังเกตว่าถึงแม้จะใช้ระดับความสูงบน Android คุณก็ยังไม่เห็น a . มากนัก
เงา. ความจริงก็คือ บน Android คุณจะไม่มีทางเข้าใกล้เอฟเฟกต์เงาที่สามารถทำได้
ผลิตบน iOS ด้วย React Native นอกกรอบ ถ้าคุณต้องมีหยดเงา-
ows บน Android ฉันแนะนำให้มองหาส่วนประกอบบน npm หรือ yarn ที่ทำหน้าที่
คุณต้องการ. ทดลองกับส่วนประกอบต่างๆ และดูว่าคุณสามารถรับ Android . ได้หรือไม่
เวอร์ชันที่คมชัดเหมือนเวอร์ชัน iOS ฉันไม่มีคำแนะนำใดๆ ฉันอยู่
ให้พ้นจากเงามืดหรือยอมรับความแตกต่าง
รหัสในบทนี้เริ่มต้นด้วยรายการ 4.20: ตัวอย่างProfileCard ที่สมบูรณ์
จากบทที่ 4 รายการ 5.1 แสดงเฉพาะการเปลี่ยนแปลงที่จำเป็นในการใช้เงาตกกระทบกับ
องค์ประกอบ คุณไม่จำเป็นต้องเพิ่มโค้ดจำนวนมากเพื่อให้ได้เงาตกกระทบบน iOS ดู
ที่รายการบนอุปกรณ์ Android และดูว่าการตั้งค่าระดับความสูงทำให้เป็นลมได้อย่างไร
ของเงา

React Native Developer
iOS
Android
John เป็น JavaScript ที่ยอดเยี่ยมจริงๆ
นักพัฒนา เขาชอบใช้ JS to
สร้างแอปพลิเคชัน React Native
สำหรับ iOS และ Android

React Native Developer
John เป็น JavaScript ที่ยอดเยี่ยมจริงๆ
นักพัฒนา เขาชอบใช้ JS to
สร้างแอปพลิเคชัน React Native
สำหรับ iOS และ Android
รูปที่ 5.4 ProfileCard บน iOS และ Android หลังจากปล่อยเงาแล้ว
เพิ่มลงในที่ใส่การ์ดและคอนเทนเนอร์รูปภาพแบบวงกลม เงาหล่นบน iOS

ถูกสร้างขึ้นโดยใช้คุณสมบัติเงาเฉพาะของ iOS: shadowColor, shadowOffset,
และเงาความทึบ บน Android จะใช้คุณสมบัติการยกระดับเพื่อพยายามสร้าง
ความลึก. มันสร้างเอฟเฟกต์เงาเพียงเล็กน้อย ซึ่งด้อยกว่าเงาที่สร้างบน iOS อย่างมาก

หน้า 50


```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { รูปภาพ, แพลตฟอร์ม, สไตล์ชีต, ข้อความ, มุมมอง} จาก 'react-native';
...
คอนเทนเนอร์การ์ด: {
...
ความสูง: 400,
...Platform.select({
ไอโอเอส: {
shadowColor: 'ดำ',
ออฟเซ็ตเงา: {
ส่วนสูง: 10
},
เงาความทึบ: 1
},
หุ่นยนต์: {
ระดับความสูง: 15
}
})
},
cardImageContainer: {
...
paddingTop: 15,
...Platform.select({
ไอโอเอส: {
shadowColor: 'ดำ',
ออฟเซ็ตเงา: {
ความสูง: 10,
},
เงาความทึบ: 1
},
หุ่นยนต์: {
ขอบกว้าง: 3,
borderColor: 'สีดำ',
```

ระดับความสูง: 15
}
})
},
...
เช่นเดียวกับการเลือกแบบอักษรในบทที่ 4 คุณใช้ฟังก์ชันPlatform.selectเพื่อนำไปใช้
รูปแบบต่างๆ ของส่วนประกอบตามแพลตฟอร์ม: iOS หรือ Android ในบางกรณี,
เช่นเดียวกับเงา แพลตฟอร์มหนึ่งอาจทำงานได้ดีกว่าอีกแพลตฟอร์มหนึ่ง แต่ใน
กรณีส่วนใหญ่รูปแบบจะทำงานเหมือนกันทั้งสองแพลตฟอร์มซึ่งเป็นข้อดีที่น่าอัศจรรย์
ผลของ React Native

5.2 การใช้การแปลงเพื่อย้าย หมุน ปรับขนาด และส่วนประกอบเอียง

จนถึงตอนนี้ รูปแบบที่เราได้พูดคุยกันส่วนใหญ่ส่งผลต่อลักษณะที่ปรากฏของ
ส่วนประกอบ คุณได้เรียนรู้วิธีตั้งค่าคุณสมบัติต่างๆ เช่น สไตล์ น้ำหนัก ขนาด และสี
ของเส้นขอบและแบบอักษร คุณใช้สีพื้นหลังและเอฟเฟกต์เงา แล้วคุณจะเห็น
วิธีจัดการลักษณะที่ปรากฏของส่วนประกอบที่สัมพันธ์กันโดยใช้
นำเข้ายูทิลิตี้แพลตฟอร์ม
ส่วนประกอบโดยทางโปรแกรม
เลือกสไตล์ตามแพลตฟอร์ม
เพิ่มเงาให้กับการ์ด
คอนเทนเนอร์ตามแพลตฟอร์ม
เพิ่มเงาให้กับ
ภาชนะรูปทรงกลม
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

123
การใช้การแปลงเพื่อย้าย หมุน ปรับขนาด และเอียงส่วนประกอบ
ระยะขอบและช่องว่างภายใน แต่เรายังไม่ได้สำรวจวิธีจัดการส่วนประกอบ
ตำแหน่งหรือการวางแนวบนหน้าจอโดยไม่ขึ้นกับสิ่งอื่นใด เป็นยังไงบ้าง
ย้ายส่วนประกอบบนหน้าจอหรือหมุนส่วนประกอบเป็นวงกลม?
คำตอบคือการเปลี่ยนแปลง React Native ให้การแปลงที่มีประโยชน์จำนวนหนึ่ง
ที่ให้คุณปรับเปลี่ยนรูปร่างและตำแหน่งของส่วนประกอบในพื้นที่ 3 มิติ คุณสามารถ

ย้ายส่วนประกอบจากตำแหน่งหนึ่งไปยังอีกตำแหน่งหนึ่ง หมุนส่วนประกอบเกี่ยวกับทั้งสาม
แกน และมาตราส่วนและส่วนเอียงในทิศทาง x และ y คนเดียว การเปลี่ยนแปลง
สามารถสร้างเอฟเฟกต์ที่น่าสนใจบางอย่างได้ แต่พลังที่แท้จริงนั้นมาจากการจัดลำดับ
มารวมกันเป็นแอนิเมชั่น
ส่วนนี้จะทำให้คุณเข้าใจอย่างลึกซึ้งเกี่ยวกับการเปลี่ยนแปลงและผลกระทบที่มีต่อ
ส่วนประกอบที่ใช้ ถ้าคุณเข้าใจสิ่งที่พวกเขาทำอย่างชัดเจน คุณจะ
สามารถเชื่อมโยงเข้าด้วยกันได้ดีขึ้นเพื่อสร้างภาพเคลื่อนไหวที่มีความหมายในภายหลัง
เปลี่ยนสไตล์ใช้เวลาอาร์เรย์ของแปลงคุณสมบัติที่กำหนดวิธีการที่จะนำไปใช้
การแปลงเป็นส่วนประกอบ ตัวอย่างเช่น ในการหมุนส่วนประกอบ 90 องศาและ
ลดขนาดลง 50% ใช้การแปลงนี้กับส่วนประกอบ:
แปลง: [{หมุน: '90deg' มาตราส่วน: .5}]
เปลี่ยนสนับสนุนสไตล์คุณสมบัติต่อไปนี้:

- มุมมอง
- translateX และtranslateY
- หมุนX , หมุน Y , และหมุนZ (หมุน)
- ขนาด, scaleX และscaleY
- เอียงX และเอียงY

5.2.1 เอฟเฟกต์ 3D พร้อมมุมมอง

เปอร์สเปคทีฟให้พื้นที่ 3 มิติขององค์ประกอบโดยส่งผลต่อระยะห่างระหว่างระนาบ z และ
ผู้ใช้งาน. ใช้กับคุณสมบัติอื่นเพื่อสร้างเอฟเฟกต์ 3 มิติ ขนาดใหญ่มุมมอง
ค่ายิ่งดัชนี z ขององค์ประกอบยิ่งมากขึ้น ซึ่งทำให้ดูใกล้ชิดกับผู้ใช้มากขึ้น ถ้า
ดัชนี z เป็นค่าลบ ยิ่งองค์ประกอบปรากฏไกลขึ้น

5.2.2 การย้ายองค์ประกอบตามแกน x และ y ด้วย translateX และแปลY

คุณสมบัติการแปลย้ายองค์ประกอบตาม x (translateX) หรือ y (translateY)
แกนจากตำแหน่งปัจจุบัน สิ่งนี้ไม่มีประโยชน์อย่างมากในการพัฒนาตามปกติเพราะคุณ
มีคุณสมบัติmargin , paddingและตำแหน่งอื่นๆ อยู่แล้ว แต่นี่กลายเป็น
มีประโยชน์สำหรับแอนิเมชั่น เพื่อย้ายส่วนประกอบข้ามหน้าจอจากตำแหน่งหนึ่งไปยัง
อื่น.
ดู Let 's ที่วิธีการที่จะย้ายตารางโดยใช้translateXและtranslateY prop- สไตล์
เออร์ตี้ ในรูปที่ 5.5 สี่เหลี่ยมจัตุรัสวางอยู่ตรงกลางของจอแสดงผลแล้วย้ายเข้ามา
แต่ละทิศของพระคาร์ดินัลสี่ทิศและทิศลำดับสี่ทิศ: NW (ซ้ายบน), N (บนสุด), NE
(ขวาบน), W (ซ้าย), E (ขวา), SW (ล่างซ้าย), S (ล่าง) และ SE (ล่างขวา)
ในแต่ละกรณี จุดศูนย์กลางของสี่เหลี่ยมจัตุรัสจะขยับ 1.5 เท่าของขนาดของสี่เหลี่ยมจัตุรัสใน x หรือ y

ทิศทางหรือทั้งสองทิศทาง

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 52

124

C HAPTER 5 จัดแต่งทรงผมอย่างล้ำลึก

เมื่อเรียนเรขาคณิต คุณพิมพ์-

แคลลลี่เห็นแกน y บวกที่วาดขึ้น

แทนที่จะลง แต่บนอุปกรณ์พกพา

แบบแผนคือการมีแกน y บวกไป

ลงจอที่สะท้อนมากที่สุด

ปฏิสัมพันธ์ทั่วไปของการเลื่อนลง

หน้าจอเพื่อดูเนื้อหาเพิ่มเติม ควบคู่ไปกับการ

ความรู้เล็กๆ น้อยๆ นั้นดูง่าย

การเคลื่อนที่ของสี่เหลี่ยมจัตุรัสกลางในรูปที่5.5

ในทิศทางบวก x และในทิศทางบวก

ทิศทาง y ส่งผลให้สี่เหลี่ยมจัตุรัสสิ้นสุด

ที่มุมล่างขวา โดยการผสมผสาน

translateX และ translateY คุณสามารถย้าย

ส่วนประกอบในทิศทางใดก็ได้ใน Carte-

เครื่องบินเซียน (ระนาบ xy)

ไม่มีคำแปลที่สอดคล้องกันสำหรับ

การเคลื่อนไหวในระนาบ z แกน z ต่อ

ตั้งฉากกับใบหน้าของอุปกรณ์ซึ่ง

หมายความว่าคุณกำลังมองตรงไปที่มัน ขนย้าย

ส่วนประกอบไปข้างหน้าหรือข้างหลังจะ

จะมองไม่เห็นโดยไม่มีการเปลี่ยนแปลงขนาดที่สอดคล้องกัน มุมมองของทรานส์

แบบฟอร์มมีวัตถุประสงค์เพื่อจัดการกับเอฟเฟกต์ภาพประเภทนี้

ในส่วนถัดไป เราจะใช้ตัวอย่างเดียวกันและเน้นที่แถวกลาง โดยที่

สี่เหลี่ยมตรงกลางถูกแปลไปทางซ้ายและทางขวา คุณจะเห็นสิ่งที่เกิดขึ้น

เมื่อคุณหมุนส่วนประกอบตามแกนทั้งสามแต่ละแกน

5.2.3 องค์ประกอบการหมุนด้วยการหมุน X, การหมุน Y และการหมุน Z (หมุน)

คุณสมบัติการหมุนทำหน้าที่เหมือนอย่างที่คิด นั่นคือจะหมุนองค์ประกอบต่างๆ
การหมุนเกิดขึ้นตามแนวแกน: x, y หรือ z จุดกำเนิดของการหมุนคือจุดศูนย์กลาง
ขององค์ประกอบก่อนที่จะใช้การแปลงใด ๆ ดังนั้นหากคุณใช้translateXหรือ
TranslateY โปรดจำไว้ว่าการหมุนจะอยู่ที่แกนที่ตำแหน่งเดิม
ชั่น จำนวนการหมุนสามารถระบุได้ทั้งองศา (องศา) หรือเรเดียน (rad)
ตัวอย่างใช้องศา:
แปลง: [{ หมุน: '45deg' }]
แปลง: [{ หมุน: '0.785398rad' }]
รูปที่ 5.6 แสดงทิศทางบวกและลบของการหมุน
สำหรับแต่ละแกน การแปลงแบบหมุนทำสิ่งเดียวกัน
ในขณะที่การหมุน Zแปลง
ลองหมุนสี่เหลี่ยมจัตุรัส 100 × 100 รอบแกน x ทีละส่วนกัน
35° ดังแสดงในรูปที่ 5.7 เส้นตรงกลางลากผ่าน
แต่ละช่องจึงง่ายต่อการดูว่าสี่เหลี่ยมหมุนอย่างไร
คุณสามารถเห็นภาพการหมุนของแกน x ในไดเร็กบวก
ในขณะที่สี่เหลี่ยมหมุนจากด้านบนเข้าสู่หน้า บอท-
ทอมเข้ามาใกล้คุณมากขึ้นในขณะที่ด้านบนเคลื่อนห่างออกไป
+X
+Y
บน
ซ้าย
ซ้าย
ล่าง
ซ้าย
ล่าง
ล่าง
ขวา
ขวา
ศูนย์กลาง
บนขวา
สูงสุด
รูปที่ 5.5 ภาพแสดงสี่เหลี่ยมจัตุรัสตรงกลาง
ถูกย้ายในแต่ละพระคาร์ดินัลทั้งสี่และ
สี่ทิศทาง: NW (บนซ้าย), N (บน)
NE (ขวาบน), W (ซ้าย), E (ขวา), SW (ล่าง
ซ้าย), S (ล่าง) และ SE (ล่างขวา)
–
–
–
Z
Y

NS
+
+
+
รูปที่ 5.6 ผลบวก
และทิศทางลบของ
การหมุนสำหรับแต่ละแกน

125
การใช้การแปลงเพื่อย้าย หมุน ปรับขนาด และเอียงส่วนประกอบ
ที่ 90° คุณกำลังดูสี่เหลี่ยมจัตุรัสที่ขอบ (เพราะไม่
มีความหนาใด ๆ คุณไม่เห็นอะไร) หลังจากที่จตุรัสมี
เมื่อหมุนผ่านจุด 90° คุณเริ่มเห็นด้านหลังของสี่เหลี่ยม ถ้า
คุณมองอย่างใกล้ชิดในรูปที่ 5.7 คุณจะเห็นป้าย "ROTATION" คือ
กลับหัวกลับหางเพราะว่าคุณกำลังมองผ่านสิ่งที่อยู่ด้านหลัง
สี่เหลี่ยม
ตัวอย่างต่อไปจะหมุนสี่เหลี่ยมจัตุรัส 100 × 100 เดียวกันเกี่ยวกับ
แกน y แทนที่จะใช้การเพิ่มทีละ 35° เพื่อสาธิต
การหมุน (ดูรูปที่ 5.8) นึกภาพด้านขวาของสี่เหลี่ยมจัตุรัสที่กำลังเคลื่อนที่
จากคุณเข้าสู่หน้า หลังจากที่จตุรัสได้หมุนไปไกลกว่า
เครื่องหมาย 90° คุณจะเห็นฉลาก “ROTATION” ผ่านส่วนประกอบ-
เน็ท เนื่องจากคุณกำลังมองผ่านด้านหลังของส่วนประกอบ
ข้อความปรากฏขึ้นย้อนกลับ
เปรียบเทียบรูปที่ 5.8 กับรูปที่ 5.7 โดยพื้นฐานแล้ว การหมุนเวียนเกี่ยวกับ
แกน y ไม่ต่างจากการหมุนรอบแกน x ฉันจัดตำแหน่ง
สี่เหลี่ยมในรูปที่ 5.8 ในแนวตั้ง เพื่อให้คุณเห็นแกนหมุนได้ง่าย
ฉันชอบนึกภาพการหมุนในแกน y โดยนึกภาพการเปิดหนังสือและ
การปิด: หากคุณกำลังเปิดหนังสือ ปกจะหมุนเป็นเชิงลบ
ทิศทาง. หากคุณกำลังปิดหนังสือ แสดงว่าคุณกำลังหมุนปกใน
ทิศทางบวก
การหมุนรอบแกน z เป็นภาพที่ง่ายที่สุด การหมุนใน
ทิศทางบวกหมุนวัตถุตามเข็มนาฬิกาและหมุน-

ในทิศทางลบหมุนสี่เหลี่ยมในทวนเข็มนาฬิกา
แฟชั่น. สำหรับตัวอย่างนี้ ดังแสดงในภาพที่ 5.9 แกนของการหมุนคือ
หมุนZ
ROTATION
0 °
35°
70 องศา
105 °
140 °
ROTATION
NS
โอ
NS
NS
NS
IO
NS
NS
โอ
NS
NS
NS
IO
NS
RO
TA
TIO
NS
รูปที่ 5.9 หมุนสี่เหลี่ยม 100 × 100 รอบแกน z ทีละส่วน
ที่ 35° การหมุนบวกคือตามเข็มนาฬิกา และการหมุนเชิงลบคือ
ทวนเข็มนาฬิกา
ROTATION
0 °
35°
70 องศา
105 °
140 °
ROTATION
ROTATION
หมุน
NS
RO
TA
Ti
บน
หมุนY
รูปที่ 5.8 การหมุน
100 × 100 ตาราง
เกี่ยวกับแกน y ใน
เพิ่มขึ้น 35 ° หลังจาก
90°, “การหมุน”

สามารถมองเห็นฉลากได้
ผ่านองค์ประกอบ
ย้อนกลับ.
ROTATION
0 °
35°
70 องศา
105 °
140 °
หมุนX
ROTATION
ROTATION
ROTATION
ROTATION
รูปที่ 5.7 การหมุนสี่เหลี่ยมจัตุรัส 100 × 100 รอบแกน x โดยเพิ่มขึ้นทีละ 35°
หลังจาก 90° คุณจะเห็นป้ายกำกับ "ROTATION" ผ่านองค์ประกอบโดยกลับหัวกลับหาง
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 54

แสดงเป็นจุดที่อยู่ตรงกลางของสี่เหลี่ยมจัตุรัส เพราะโดยพื้นฐานแล้วแกน z คือเส้นตรงของคุณ
ของสายตา; มันตรงเข้าสู่หน้าจอ
หวังว่าตอนนี้จะค่อนข้างชัดเจนว่าการหมุนจะเปลี่ยนไปอย่างไร ความเข้าใจ
ทิศทางที่การหมุนบวกและลบส่งผลต่อวัตถุน่าจะเป็น
ส่วนที่ซับซ้อนที่สุด แต่เมื่อคุณเริ่มรวมการแปลงอื่น ๆ ร่วมกับ
การหมุนคุณอาจประหลาดใจกับผลลัพธ์ จำไว้ว่าคุณสมบัติการแปลงรูป
เป็นอาร์เรย์ของการแปลงร่าง จึงสามารถจัดหาการแปลงหลายรายการพร้อมกันได้ และลำดับ
เรื่อง! การระบุการแปลงและการสลับลำดับขององค์ประกอบในอาร์เรย์ will
ให้ผลลัพธ์ที่แตกต่างกัน
มาดูกันว่าการเปลี่ยนลำดับการระบุการแปลงส่งผลกระทบกับ
เค้าโครงสุดท้าย ลองใช้การแปลงที่แตกต่างกันสามแบบกับสี่เหลี่ยมจัตุรัส: แปลในทิศทาง y
50 คะแนน แปลในทิศทาง x 150 คะแนน แล้วหมุนสี่เหลี่ยม 45° รูป 5.10
ระบุการแปลงตามลำดับที่อธิบายไว้ ตำแหน่งเดิม/ก่อนหน้าของ
สี่เหลี่ยมจัตุรัสมีเส้นขอบประ และตำแหน่งใหม่ของสี่เหลี่ยมมีโครงร่างทึบ ดังนั้นคุณ
สามารถดูได้ว่าการเปลี่ยนแปลงส่งผลต่อตำแหน่งและทิศทางของสี่เหลี่ยมจัตุรัสเดิมอย่างไร
ผลลัพท์ในรูป 5.10 สวยมากตามคาด แต่จะเกิดอะไรขึ้นถ้าสมัคร

การหมุนหลังจากเคลื่อนสี่เหลี่ยมจัตุรัสไปในทิศทาง y? ดูรูป 5.11 และหา
ห๊ะ เกิดอะไรขึ้น? สี่เหลี่ยมจตุรัสปิดหน้าจออย่างสมบูรณ์หลังจากการแปลง
mations ถูกนำไปใช้! มันอาจจะไม่ชัดเจนในทันทีว่าเกิดอะไรขึ้นซึ่งเป็นสาเหตุ
รูปที่ 5.11 มีคำอธิบายประกอบด้วยการวางแนวแกนใหม่
หลังจากการหมุน แกน +x- และ +y- จะไม่ถูกจัดวางในแนวตั้งและแนวนอนอีกต่อไป
บนหน้าจอในแนวตั้ง: โดยจะหมุนไป 45° เมื่อการแปลง translateXเป็น
ใช้สี่เหลี่ยมจัตุรัสถูกย้าย 150 จุดในทิศทาง +x แต่ทิศทาง +x คือ
ตอนนี้ทำมุม 45° จากแกน x เดิม
ส่วนถัดไปแสดงอีกแง่มุมที่น่าสนใจของการแปลงแบบหมุน
แปลง: [{translateY: 50},{translateX: 150},{หมุน: '45deg'}]
สี่เหลี่ยมเดิม
ขั้นตอนที่ 1: {translateY: 50}
ขั้นตอนที่ 2: {translateX: 150}
ขั้นตอนที่ 3: {หมุน: '45deg'}
การวางแนวแกนใหม่
+X
รูปที่ 5.10 การใช้การแปลง: [{translateY: 50},{translateX:
150},{หมุน: '45deg'}] เป็นสี่เหลี่ยมจัตุรัสเดิม
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 55

5.2.4 การตั้งค่าการมองเห็นเมื่อหมุนองค์ประกอบมากกว่า 90°

หากคุณมองย้อนกลับไปที่ตัวเลข 5.7 และ 5.8 เมื่อคุณหมุนสี่เหลี่ยมจัตุรัสรอบแกน x หรือ y
และเกินจุด 90° คุณยังเห็นข้อความที่อยู่ด้านหน้าของ
สี่เหลี่ยม. backfaceVisibilityสั่งสถานที่ให้บริการไม่ว่าจะเป็นองค์ประกอบที่สามารถมองเห็นได้เมื่อ
องค์ประกอบถูกหมุนมากกว่า 90° คุณสมบัตินี้สามารถตั้งค่าเป็น'มองเห็นได้'หรือ
'ซ่อน' . คุณสมบัตินี้ไม่ใช่การแปลงรูปแบบ แต่ให้ความสามารถในการซ่อนหรือแสดง
องค์ประกอบเมื่อดูใบหน้าด้านหลังของวัตถุ
backfaceVisibilityคุณสมบัติเริ่มต้นที่'มองเห็น'แต่ถ้าคุณเปลี่ยนBack-
faceVisibility to 'hidden' คุณจะไม่เห็นองค์ประกอบเลยทันทีที่ส่วนประกอบ
หมุนมากกว่า 90° ในทิศทาง x หรือ y ในรูปที่ 5.7 และ 5.8 กำลังสอง

ที่สอดคล้องกับการหมุน 105 °และ 140 °จะหายไป ถ้ามันฟังดูสับสน-
ing, ดูรูปที่ 5.12.

2

5

4

Cube: backfaceVisibility: 'มองเห็นได้'
Cube: backfaceVisibility: 'ซ่อน'
รูปที่ 5.12 การสาธิตการตั้งค่าคุณสมบัติ backfaceVisibility
เพื่อ 'ซ่อน' ซ่อนองค์ประกอบที่หมุนเกิน 90° ลูกบาศก์ทางซ้าย
แสดงใบหน้า 2, 4 และ 5 ซึ่งทั้งหมดได้หมุนไป 180° ลูกบาศก์ทางด้านขวามี
ซ่อนใบหน้าเหล่านั้น
แปลง: [{translateY: 50},{rotate: '45deg'},{translateX: 150}]
สี่เหลี่ยมเดิม
ขั้นตอนที่ 1: {translateY: 50}
ขั้นตอนที่ 2: {หมุน: '45deg'}
ขั้นตอนที่ 3: {translateX: 150}
การวางแนวแกนใหม่
+X
รูปที่ 5.11 การใช้การแปลง: [{translateY: 50},{rotate:
'45deg'},{translateX: 150}] เป็นสี่เหลี่ยมจัตุรัสเดิม หมุนสี่เหลี่ยม
เปลี่ยนการวางแนวของแกน x และ y ดังนั้นเมื่อแปลสี่เหลี่ยมจัตุรัส 150
ชี้ไปในทิศทาง +x เลื่อนแนวทแยงมุมลงและออกจากวิวพอร์ต
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 56

ในรูป คุณสามารถเห็นผลของการตั้งค่าbackfaceVisibilityเป็น'hidden'ได้อย่างง่ายดาย
นอกจากนี้ยังง่ายต่อการดูว่าพฤติกรรมนี้จะเป็นประโยชน์ระหว่างแอนิเมชั่นอย่างไร เมื่อ
ใบหน้าของลูกบาศก์หมุนไปจนลับตา คุณต้องการให้ซ่อนมันไว้

5.2.5 การปรับขนาดวัตถุบนหน้าจอด้วยมาตราส่วน มาตราส่วน X และมาตราส่วน Y

ส่วนนี้พูดถึงการปรับขนาดวัตถุบนหน้าจอ มีประโยชน์หลายอย่าง
สำหรับการปรับขนาดและรูปแบบต่างๆ ที่ใช้ประโยชน์จากความสามารถของมัน ตัวอย่างเช่น มาตราส่วน-
ing สามารถใช้เพื่อสร้างภาพขนาดย่อของวัตถุ คุณเคยเห็นสิ่งนี้ในแอปพลิเคชั่นมากมาย
ผู้ใช้แตะภาพขนาดย่อ และภาพเคลื่อนไหวจะค่อยๆ ปรับขนาดวัตถุกลับให้เต็ม
ขนาด. เป็นเทคนิคการเปลี่ยนภาพทั่วไปที่ให้เอฟเฟกต์ภาพที่สวยงาม
คุณจะได้เรียนรู้พื้นฐานของการปรับขนาดวัตถุแล้วใช้ทักษะเหล่านั้นเพื่อสร้างนิ้วหัวแม่มือ-
เล็บของProfileCardที่เปิดขึ้นขนาดเต็มเมื่อกด ต่อมาในบทนี้เ-
cusses flexbox และวิธีใช้เพื่อจัดการรูปขนาดย่อของProfileCard
ในอินเทอร์เฟซแกลเลอรี ซึ่งคุณสามารถกดโปรไฟล์เพื่อดูรายละเอียดเพิ่มเติมได้
มาตราส่วนคูณขนาดขององค์ประกอบด้วยจำนวนที่ส่งไป ค่าเริ่มต้น
เป็น 1. ในการทำให้องค์ประกอบดูใหญ่ขึ้น ให้ส่งค่าที่มากกว่า 1 เพื่อทำมัน
มีขนาดเล็กลง ส่งผ่านค่าที่น้อยกว่า 1
องค์ประกอบนอกจากนี้ยังสามารถปรับขนาดตามแกนเดียวโดยใช้scaleXหรือscaleY ขนาดX
ยืดองค์ประกอบในแนวนอนตามแนวแกน x และมาตราส่วนYยืดองค์ประกอบ
ในแนวตั้งตามแนวแกน y มาสร้างสี่เหลี่ยมสองสามอันเพื่อแสดงผลกระทบของการปรับขนาด: ดู
รูปที่ 5.13.
ไม่มีอะไรผิดปกติเกิดขึ้น การปรับขนาดวัตถุนั้นค่อนข้างตรงไปตรงมา รายการ 5.2
แสดงให้เห็นว่ามันง่ายแค่ไหน

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { StyleSheet, Text, View} จาก 'react-native';
```

รูปที่ 5.13 ตัวอย่างการปรับขนาดแปลงสี่เหลี่ยมจัตุรัสเดิมอย่างไร ทั้งหมด
สี่เหลี่ยมเริ่มมีขนาดและรูปร่างเท่ากับ A ซึ่งมีมาตราส่วนเริ่มต้น
ของ 1. B ปรับขนาดสี่เหลี่ยมจัตุรัส 0.5 แล้วย่อให้เล็กลง C ปรับขนาดสี่เหลี่ยมจัตุรัสเป็น 2
ขยายมัน D ใช้ scaleX แปลงสี่เหลี่ยมจัตุรัสตามแนวแกน x โดย
3x. E ใช้สเกลY เปลี่ยนสี่เหลี่ยมจัตุรัสตามแนวแกน y 1.5x

การใช้การแปลงเพื่อย้าย หมุน ปรับขนาด และเอียงส่วนประกอบ

```
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<รูปแบบตัวอย่าง={{}}>A,1</Example>
<รูปแบบตัวอย่าง={{แปลง: [{มาตราส่วน: 0.5}]}}>B,0.5</ตัวอย่าง>
<รูปแบบตัวอย่าง={{แปลง: [{มาตราส่วน: 2}]}}>C,2</ตัวอย่าง>
<รูปแบบตัวอย่าง={{แปลง: [{ขนาดX: 3}]}}>D,X3</ตัวอย่าง>
<รูปแบบตัวอย่าง={{แปลง: [{สเกลY: 1.5}]}}>E,Y1.5</ตัวอย่าง>
</ดู>
);
}
}
const ตัวอย่าง = (อุปกรณ์ประกอบฉาก) => (
<ดู style={[styles.example,props.style]}>
<ข้อความ>
{อุปกรณ์ประกอบฉาก.เด็ก}
</Text>
</ดู>
);
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ขอบบน: 75,
alignItems: 'ศูนย์',
ดิ้น: 1
},
ตัวอย่าง: {
ความกว้าง: 50,
ความสูง: 50,
ขอบกว้าง: 2,
ระยะขอบ: 15,
alignItems: 'ศูนย์',
justifyContent: 'ศูนย์กลาง'
},
});
```

5.2.6 การใช้การแปลงมาตราส่วนเพื่อสร้างภาพขนาดย่อของ ProfileCard

เมื่อคุณได้เห็นการใช้งานจริงแล้ว ลองใช้เทคนิคนี้เพื่อสร้างภาพขนาดย่อของProfileCard ปกติคุณจะเคลื่อนไหวในสิ่งที่ฉันกำลังจะแสดงให้คุณเห็นเพื่อหลีกเลี่ยงริบหรี่ แต่มาดูวิธีการใช้ scaling ในทางปฏิบัติกัน รูปที่ 5.14 แสดงขนาดเล็กเวอร์ชันย่อของส่วนประกอบProfileCardซึ่งเป็นภาพขนาดย่อ หากคุณกดภาพขนาดย่อ ส่วนประกอบจะกลับสู่ขนาดเต็ม หากคุณกดส่วนประกอบขนาดเต็ม

มันจะยุบกลับลงมาเป็นมุมมองภาพขนาดย่อ
เริ่มต้นด้วยรหัสจากรายการ 5.1 ตราบใดที่สไตล์ดำเนินไป คุณต้องเพิ่มใหม่เพียงอันเดียว
สไตล์ที่จะทำการแปลงมาตราส่วนจากขนาดเต็มเป็นภาพขนาดย่อ รหัสที่เหลือ
จัดเรียงชิ้นส่วนของส่วนประกอบใหม่ให้เป็นโครงสร้างที่นำกลับมาใช้ใหม่ได้มากขึ้นและให้
ความสามารถในการสัมผัสเพื่อจัดการกับเหตุการณ์onPress
ค่าเริ่มต้น 50 × 50 ตาราง
โดยไม่ต้องใช้มาตราส่วน
ปรับขนาดสี่เหลี่ยมเริ่มต้น
0.5 ลดขนาดลง
ปรับขนาดสี่เหลี่ยมเริ่มต้นเป็น 2 ทำให้ใหญ่ขึ้น
ตาชั่ง
สี่เหลี่ยมเริ่มต้น
เฉพาะใน x
ทิศทาง,
ยึดมัน
แนวนอน
ตาชั่ง
สี่เหลี่ยมเริ่มต้น
เฉพาะใน y
ทิศทาง,
ยึดมัน
แนวตั้ง
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 58


```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า PropTypes จาก 'prop-types';
นำเข้าการอัปเดตจาก 'ตัวช่วยที่ไม่สามารถเปลี่ยนแปลงได้';
นำเข้า { รูปภาพ, แพลตฟอร์ม, สไตล์ชีต, ข้อความ,
TouchableHighlight, View} จาก 'react-native';
const userImage = ต้องการ ('./user.png');
```

ข้อมูล const = [{
ภาพ: userImage,
ชื่อ 'จอห์น โด'
อาชีพ: 'React Native Developer',
คำอธิบาย: 'John เป็นนักพัฒนา Javascript ที่ยอดเยี่ยมจริงๆ ' +
'เขาชอบใช้ JS เพื่อสร้างแอปพลิเคชัน React Native' +
'สำหรับ iOS และ Android'
แสดงภาพขนาดย่อ: จริง
}
];
const ProfileCard = (อุปกรณ์ประกอบฉาก) => {
const { ภาพ, ชื่อ, อาชีพ,
คำอธิบาย onPress, showThumbnail } = อุปกรณ์ประกอบฉาก;
ให้ containerStyles = [styles.cardContainer];
จอห์น โด
React Native Developer
John เป็น JavaScript ที่ยอดเยี่ยมจริงๆ
นักพัฒนา เขาชอบใช้ JS to
สร้างแอปพลิเคชัน React Native
สำหรับ iOS และ Android
จอห์น โด
React Native Developer
John เป็น JavaScript ที่ยอดเยี่ยมจริงๆ
นักพัฒนา เขาชอบใช้ JS to
สร้างแอปพลิเคชัน React Native
สำหรับ iOS และ Android
คลิกเพื่อยุบ
ลงในภาพขนาดย่อ
คลิกเพื่อขยาย
รูปที่ 5.14 การปรับขนาด ProfileCard ขนาดเต็มลง 80% เป็น a
ภาพขนาดย่อ การกดรูปขนาดย่อจะคืนค่า ProfileCard เป็น
ขนาดดั้งเดิมและการกดส่วนประกอบขนาดเต็มจะยุบส่วนประกอบ
ลงในภาพขนาดย่อ
PropTypes ให้คุณระบุสิ่งที่
คุณสมบัติ ProfileCard
ส่วนประกอบสามารถยอมรับได้
NS
ไม่เปลี่ยนรูป
ผู้ช่วย
การทำงาน
อัปเดตกันเถอะ
คุณอัปเดต a
ชิ้นเฉพาะ

ของ
ส่วนประกอบ
สถานะ.
ส่วนประกอบ TouchableHighlight
เปิดใช้งานการประมวลผลแบบสัมผัส
แยกองค์ประกอบข้อมูลแล้ว
เพื่อสรุปองค์ประกอบ
ส่วนประกอบ ProfileCard อยู่ในขณะนี้
แยกออกจากรหัสแอป
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 59

131
การใช้การแปลงเพื่อย้าย หมุน ปรับขนาด และเอียงส่วนประกอบ

```
ถ้า (showThumbnail) {
containerStyles.push(styles.cardภาพขนาดย่อ);
}
กลับ (
<TouchableHighlight onPress={onPress}>
<ดู style={[containerStyles]}>
<ดู style={styles.cardImageContainer}>
<Image style={styles.cardImage} แหล่งที่มา={image}/>
</ดู>
<ดู>
<Text style={styles.cardName}>
{ชื่อ}
</Text>
</ดู>
<ดู style={styles.cardOccupationContainer}>
<Text style={styles.cardOccupation}>
{อาชีพ}
</Text>
</ดู>
<ดู>
<Text style={styles.cardDescription}>
{คำอธิบาย}
</Text>
</ดู>
</ดู>
```

```
</TouchableHighlight>
)
};
ProfileCard.propTypes = {
ภาพ: PropTypes.number.isRequired,
ชื่อ: PropTypes.string.isRequired,
อาชีพ: PropTypes.string.isRequired,
คำอธิบาย: PropTypes.string.isRequired,
showThumbnail: PropTypes.bool.isRequired,
onPress: PropTypes.func.isRequired
};
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
ตัวสร้าง (อุปกรณ์ประกอบฉากบริบท) {
สุดยอด(อุปกรณ์ประกอบฉากบริบท);
this.state = {
data: data
}
}
handleProfileCardPress = (ดัชนี) => {
const showThumbnail = !this.state.data[ดัชนี].showThumbnail;
this.setState({
ข้อมูล: อัปเดต (this.state.data,
{[ดัชนี]: {showThumbnail: {$set: showThumbnail}}})
});
};
เรนเดอร์ () {
const list = this.state.data.map (ฟังก์ชัน (รายการ, ดัชนี) {
ถ้า showThumbnail เป็นจริง the
ส่วนประกอบถูกลดขนาดลง 80%
กระบวนการกดเพื่อย่อให้เล็กสุด
และเพิ่มองค์ประกอบให้สูงสุด
สถานะส่วนประกอบยังคงอยู่ใน
ส่วนประกอบแอปที่มีลำดับสูงกว่า
ฟังก์ชันตัวจัดการto
ดำเนินการเกี่ยวกับข่าวเหตุการณ์
รายการ (อาร์เรย์) ของส่วนประกอบ ProfileCard
```

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

```
const { ภาพ, ชื่อ, อาชีพ, คำอธิบาย, แสดงภาพขนาดย่อ } = รายการ;
ส่งคืน <ProfileCard key={'card-' + index}
ภาพ={ภาพ}
ชื่อ={ชื่อ}
อาชีพ={อาชีพ}
คำอธิบาย={คำอธิบาย}
onPress={this.handleProfileCardPress.bind(นี่ ดัชนี)}
showThumbnail={showThumbnail}/>
}, นี้);
กลับ (
<ดู style={styles.container}>
{รายการ}
</ดู>
);
}
}
...
การ์ดรูปขนาดย่อ: {
แปลง: [{ขนาด: 0.2}]
},
...
```

ด้วยการจัดระเบียบโครงสร้างของส่วนประกอบใหม่ คุณสามารถจัดการกับการเพิ่มได้มากขึ้น
ส่วนประกอบ ProfileCard ของแอปพลิเคชัน ในส่วนที่ 5.3 คุณจะต้องเพิ่มโปรไฟล์-
การ์ดและดูวิธีการจัดระเบียบให้เป็นเลย์เอาต์ของแกลเลอรี่

5.2.7 องค์ประกอบเบ้ตามแกน x และ y ที่มีความเบ้X และ ความเบ้Y

ก่อนที่เราจะออกจากการแปลงและพูดคุยเกี่ยวกับเลย์เอาต์ มาดูskewXและskewYทรานส์-
การก่อตัว ในซอร์สโค้ดที่สร้างคิวบ์สำหรับbackfaceVisibility
ตัวอย่างที่แสดงในรูปที่ 5.12 (github บทที่5/figures/Figure-5.12-BackfaceVisibility)
คุณจะเห็นได้ว่าการบิดสี่เหลี่ยมนั้นจำเป็นต่อการสร้างสามมิติ
ส่งผลกระทบต่อใบหน้าลูกบาศก์ มาพูดคุยกันว่าskewXและskewYทำอะไรกัน ดังนั้นเมื่อคุณสำรวจ
ซอร์สโค้ดอย่างละเอียด คุณจะเข้าใจสิ่งที่คุณเห็น
skewXคุณสมบัติ skews องค์ประกอบตามแนวแกน x ในทำนองเดียวกันเสาเอียง -
erty เอียงองค์ประกอบตามแกน y รูปที่ 5.15 แสดงผลการเอียง a
สี่เหลี่ยมดังนี้:

- Square A ไม่มีการเปลี่ยนแปลงใดๆ
- สี่เหลี่ยม B เอียงไปตามแกน x 45°
- สี่เหลี่ยม C เอียงตามแนวแกน x -45°
- สี่เหลี่ยม D เอียงไปตามแกน y 45°

▪ สี่เหลี่ยม E เอียงตามแกน y ไป -45°

เช่นเดียวกับการปรับขนาด การบิดเบือนองค์ประกอบนั้นค่อนข้างง่าย: ระบุมุมและระบุ
แกน รายการถัดไปให้รายละเอียดทั้งหมด

หมายเหตุในขณะที่เขียน การแปลงskewX ทำงานไม่ถูกต้องบน
แอนดรอยด์

แสดงรายการในคอนเทนเนอร์โดยรวม

รูปแบบรูปย่อของการ์ดลดลง

ขนาดส่วนประกอบ 80%

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 61

133

การใช้การแปลงเพื่อย้าย หมุน ปรับขนาด และเอียงส่วนประกอบ

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { StyleSheet, Text, View} จาก 'react-native';
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<รูปแบบตัวอย่าง={{}}>A</Example>
<รูปแบบตัวอย่าง={{แปลง: [{skewX: '45deg'}]}}>
B X45
</ตัวอย่าง>
<รูปแบบตัวอย่าง={{แปลง: [{skewX: '-45deg'}]}}>
ซี เอ็กซ์-45
</ตัวอย่าง>
<รูปแบบตัวอย่าง={{แปลง: [{skewY: '45deg'}]}}>
DY45
</ตัวอย่าง>
<รูปแบบตัวอย่าง={{แปลง: [{skewY: '-45deg'}]}}>
อี วาย-45
</ตัวอย่าง>
</ดู>
);
```

เอียงสี่เหลี่ยม 45 °

ตามแนวแกน X

เอียงสี่เหลี่ยม –45°
ตามแนวแกน x
เอียงสี่เหลี่ยม 45 °
ตามแนวแกน y
เอียงสี่เหลี่ยม –45°
ตามแนวแกน y
รูปที่ 5.15 ตัวอย่างของการเอียงสี่เหลี่ยมตาม x-
และแกน y บน iOS Square A ไม่มีการแปลงที่ใช้
สี่เหลี่ยม B เอียงไปตามแกน x 45° สี่เหลี่ยม C คือ
เอียงตามแกน x ไป -45° สี่เหลี่ยม D เบ้ตาม
แกน y 45° และสี่เหลี่ยม E จะเบ้ไปตามแกน y
โดย –45°
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 62


```
}
}
const ตัวอย่าง = (อุปกรณ์ประกอบฉาก) => (
<ดู style={[styles.example,props.style]}>
<ข้อความ>
{อุปกรณ์ประกอบฉาก.เด็ก}
</Text>
</ดู>
);
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ขอบบน: 50,
alignItems: 'ศูนย์',
ดิ้น: 1
},
ตัวอย่าง: {
ความกว้าง: 75,
ความสูง: 75,
ขอบกว้าง: 2,
ระยะขอบ: 20,
alignItems: 'ศูนย์',
justifyContent: 'ศูนย์กลาง'
```

},
});

5.2.8 ประเด็นสำคัญของการเปลี่ยนแปลง

เราได้กล่าวถึงแนวคิดที่เปลี่ยนแปลงมากมายในส่วนนี้! บางคนค่อนข้างจะค่อนข้าง
เรียบง่าย ในขณะที่คนอื่นอาจนึกภาพไม่ออกในตอนแรก ฉันไม่ได้แสดงข้อสอบมาก-
ที่รวมการแปลงเข้าด้วยกัน ดังนั้นคุณจึงสามารถมุ่งเน้นไปที่สิ่งที่แต่ละการแปลงทำ ผม
ขอแนะนำให้คุณใช้ตัวอย่างใด ๆ และรวมถึงการเปลี่ยนแปลงเพิ่มเติมเพื่อ
ทดลองและดูว่าเกิดอะไรขึ้น
ในบทที่ 7 เมื่อเราพูดถึงแอนิเมชั่น คุณจะเห็นว่าการแปลงร่างสามารถทำได้อย่างไร
สิ่งต่าง ๆ มีชีวิตชีวา ในตอนนี้ ให้นำประเด็นสำคัญเหล่านี้ออกไป:

- จุดกำเนิดของแกน x และ y อยู่ที่ด้านซ้ายบน หมายถึง ทิศทางบวกของ y
อยู่ที่หน้าจอ คุณเห็นสิ่งนี้ด้วยตำแหน่งที่แน่นอนในบทที่แล้ว
แต่มีแนวโน้มว่าจะตรงกันข้ามกับสิ่งที่คุณคุ้นเคย ซึ่งทำให้ยากต่อการให้เหตุผล
เกี่ยวกับสิ่งที่การเปลี่ยนแปลงจะทำ
- ต้นกำเนิดของการหมุนและการแปลจะอยู่ที่ตำแหน่งเดิมขององค์ประกอบเสมอ
คุณไม่สามารถแปลวัตถุในทิศทาง x หรือ y แล้วหมุนรอบวัตถุใหม่
จุดศูนย์กลาง

การแปลงรูปแบบเป็นวิธีที่ดีในการเคลื่อนย้ายส่วนประกอบต่างๆ ไปรอบๆ หน้าจอ แต่คุณจะไม่ทำ
ใช้เป็นประจำทุกวัน ส่วนใหญ่แล้วคุณจะใช้ Yoga ซึ่งเป็นเอ็นจิ้นการจัดวางที่ง่าย
ระบุข้อกำหนดเว็บ flexbox ของ W3C ส่วนใหญ่ ในตอนต่อไปเราจะพูดถึง
รายละเอียดการใช้งาน flexbox ของ Yoga

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 63

135
ใช้ exbox เพื่อจัดวางส่วนประกอบ

5.3 การใช้ flexbox เพื่อจัดวางส่วนประกอบ

Flexbox คือการนำเลย์เอาต์ไปใช้ที่ React Native ใช้เพื่อจัดเตรียมวิธีที่มีประสิทธิภาพ
สำหรับผู้ใช้ในการสร้าง UI และควบคุมตำแหน่ง React Native flexbox นำไปใช้-
อิงตามข้อกำหนดเว็บ W3C flexbox แต่ไม่ได้แชร์ API 100%
มีจุดมุ่งหมายเพื่อให้คุณมีวิธีการง่ายๆ ในการให้เหตุผล จัดตำแหน่ง และกระจายพื้นที่ระหว่างรายการต่างๆ
ในเลย์เอาต์ แม้จะไม่ทราบขนาดหรือไดนามิกก็ตาม

หมายเหตุเลย์เอาต์ Flexbox ใช้ได้เฉพาะกับส่วนประกอบ View

คุณเคยเห็น flexbox ใช้ในตัวอย่างมากมายแล้ว มันทรงพลังและทำให้
การจัดวางสิ่งของได้ง่ายกว่าวิธีอื่นที่ยากจะไม่ใช้
คุณจะได้รับประโยชน์อย่างมากจากการใช้เวลาทำความเข้าใจเนื้อหาในส่วนนี้ ที่นี่
เป็นคุณสมบัติการจัดตำแหน่งที่ใช้ในการควบคุมเค้าโครง flexbox: flex , flexDirection ,
justifyContent , alignItems , alignSelf และflexWrap

5.3.1 การเปลี่ยนขนาดของส่วนประกอบด้วย flex

ดิ้นคุณสมบัติระบุความสามารถขององค์ประกอบในการเปลี่ยนแปลงขนาดของมันในการกรอกข้อมูล
พื้นที่ของคอนเทนเนอร์ที่มีอยู่ ค่านี้สัมพันธ์กับคุณสมบัติflexที่ระบุสำหรับ
รายการที่เหลือในภาชนะเดียวกัน
ก 50%
ค 33%
อี 25%
ฉ 75%
ง 66%
ข 50%
รูปที่ 5.16 ตัวอย่างเลย์เอาต์สามตัวอย่างโดยใช้คุณสมบัติ flex ด้านบน
ตัวอย่างคือ 1:1 โดย A = {flex: 1} และ B = {flex: 1} ส่งผลให้แต่ละ
ใช้พื้นที่ 50% ตัวอย่างตรงกลางคือ 1:2 โดยที่ C = {flex: 1}
และ D = {flex: 2} ส่งผลให้ C ใช้พื้นที่ 33% และ D ใช้พื้นที่
66%. ตัวอย่างด้านล่างคือ 1:3 โดย E = {flex: 1} และ F = {flex: 3}
ส่งผลให้ E ใช้พื้นที่ 25% และ F ใช้พื้นที่ 75%

หน้า 64

หากคุณมีองค์ประกอบView ที่มีความสูง 300 และความกว้าง 300 และView ย่อย
องค์ประกอบที่มีคุณสมบัติของflex: 1จากนั้นมุมมองลูกจะเติม parent . ให้สมบูรณ์
ดู. ถ้าคุณตัดสินใจที่จะเพิ่มองค์ประกอบลูกอีกด้วยดิ้นทรัพย์สินของเฟล็กซ์: 1 ,
แต่ละมุมมองจะใช้พื้นที่เท่ากันในคอนเทนเนอร์หลัก ดิ้นจำนวนเป็นเพียง
สำคัญเมื่อเทียบกับรายการดิ้นอื่น ๆที่ใช้พื้นที่เดียวกัน
อีกวิธีหนึ่งในการดูสิ่งนี้คือการคิดว่าคุณสมบัติเฟล็กซ์เป็นเปอร์เซ็นต์
ตัวอย่างเช่น หากคุณต้องการให้ส่วนประกอบย่อยเพิ่มขึ้น 66.6% และ 33.3% ให้คำนึงถึง
คุณสามารถใช้flex:66และflex:33ได้อย่างคล่องแคล่ว แทนที่จะflex:66และflex:33คุณสามารถ
ระบุflex:2และflex:1และรับเอฟเฟกต์เลย์เอาต์เดียวกัน

เพื่อให้เข้าใจวิธีการทำงานดีขึ้น มาดูตัวอย่างที่แสดงในรูปที่ 5.16
สิ่งเหล่านี้ทำได้โดยง่ายโดยการตั้งค่าเฟล็กซ์ที่เหมาะสมบนองค์ประกอบแต่ละรายการ
เมนชั่น รายการต่อไปนี้แสดงขั้นตอนที่จำเป็นในการสร้างเค้าโครงดังกล่าว

```
…
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<ดู style={[styles.flexContainer]}>
<example style={[{flex: 1},styles.darkgrey]}>A 50%</Example>
<รูปแบบตัวอย่าง={[{flex: 1}]}>B 50%</Example>
</ดู>
<ดู style={[styles.flexContainer]}>
<example style={[{flex: 1},styles.darkgrey]}>C 33%</Example>
<รูปแบบตัวอย่าง={{flex: 2}}>D 66%</Example>
</ดู>
<ดู style={[styles.flexContainer]}>
<example style={[{flex: 1},styles.darkgrey]}>E 25%</Example>
<รูปแบบตัวอย่าง={{flex: 3}}>F 75%</Example>
</ดู>
</ดู>
);
}
…
```

5.3.2 การกำหนดทิศทางของดิ้นด้วย flexDirection

ในตัวอย่างก่อนหน้านี้ รายการในคอนเทนเนอร์แบบยืดหยุ่นจะจัดวางในคอลัมน์ (แกน y)
แปลว่า บนลงล่าง A ซ้อนกันบน B, C ซ้อนกันบน D และ E วางซ้อนกันบน F.
การใช้คุณสมบัติflexDirectionคุณสามารถเปลี่ยนแกนหลักของเค้าโครงและ
จึงเปลี่ยนทิศทางการจัดวาง flexDirectionถูกนำไปใช้กับพาเรนต์
มุมมองที่มี
ทั้งหมดที่จำเป็นเพื่อให้ได้เลย์เอาต์ในรูปที่ 5.17 คือการเพิ่มโค้ดบรรทัดเดียวลงใน
flexContainerสไตล์ซึ่งเป็นที่เก็บหลักสำหรับแต่ละตัวอย่างซับซ้อน
ponents การเปลี่ยนflexDirectionบนคอนเทนเนอร์นี้ส่งผลต่อเลย์เอาต์ของ flex . ทั้งหมด
เด็ก. เพิ่มflexDirection: 'แถว'ให้กับสไตล์ และดูว่ามันเปลี่ยนเลย์เอาต์อย่างไร
รายการมีค่าดิ้นเหมือนกัน
ดังนั้นพวกเขาจึงใช้ปริมาณเท่ากันของ
พื้นที่ในคอนเทนเนอร์หลัก
C ใช้พื้นที่ 1/3 ของพื้นที่ทั้งหมดและ
D ใช้พื้นที่ 2/3 ของพื้นที่ทั้งหมด

E ใช้พื้นที่ 1/4 ของพื้นที่ทั้งหมดและ
F ใช้พื้นที่ 3/4 ของพื้นที่ทั้งหมด
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

137
ใช้ exbox เพื่อจัดวางส่วนประกอบ

```
flexContainer: {
ความกว้าง: 150,
ความสูง: 150,
ขอบกว้าง: 1,
ระยะขอบ: 10,
flexDirection: 'แถว'
},
```

ตอนนี้องค์ประกอบย่อยปรากฏจากซ้ายไปขวา flexDirectionมีสองตัวเลือก:
'แถว' และ'คอลัมน์' ค่าเริ่มต้นคือคอลัมน์ '' หากคุณไม่ได้ระบุ flexDirection คุณสมบัติเนื้อหาจะออกมาวางในคอลัมน์ คุณสมบัตินี้เป็นสิ่งที่คุณจะใช้จำนวนมากในการพัฒนาแอพใน React Native ดังนั้นสิ่งสำคัญคือต้องเข้าใจมันและยืนว่ามันทำงานอย่างไร

5.3.3 การกำหนดวิธีการใช้พื้นที่รอบๆ ส่วนประกอบด้วย justifyContent

เมื่อใช้คุณสมบัติflexคุณสามารถระบุได้ว่าแต่ละส่วนประกอบใช้พื้นที่เท่าใดในคอนเทนเนอร์หลัก แต่ถ้าคุณไม่พยายามกินพื้นที่ทั้งหมดล่ะ ทำอย่างไร คุณใช้ flexbox เพื่อจัดวางส่วนประกอบโดยใช้ขนาดดั้งเดิมหรือไม่ justifyContent กำหนดวิธีการกระจายพื้นที่ระหว่างและรอบ ๆ รายการแบบยืดหยุ่นตามแกนหลักของภาชนะ (ทิศทางงอ) justifyContentถูกประกาศบนคอนเทนเนอร์หลัก มีห้าตัวเลือก:

- ศูนย์ทำให้ลูกอยู่กึ่งกลางภายในคอนเทนเนอร์หลัก ฟรี
พื้นที่กระจายทั้งสองด้านของกลุ่มลูกที่กระจุกตัว
- flex-start จัดกลุ่มส่วนประกอบที่จุดเริ่มต้นของคอลัมน์ flex หรือแถวขึ้นอยู่กับค่าสิ่งที่ได้รับมอบหมายให้flexDirection flex-startคือค่าเริ่มต้นสำหรับjustifyContent
- flex-end ทำหน้าที่ตรงกันข้าม: จัดกลุ่มรายการต่างๆ เข้าด้วยกันที่ส่วนท้ายของ

คอนเทนเนอร์.

▪ ช่องว่างรอบ ๆพยายามกระจายพื้นที่รอบ ๆ แต่ละองค์ประกอบเท่า ๆ กัน อย่า
สับสนกับการกระจายองค์ประกอบอย่างสม่ำเสมอในภาชนะ พื้นที่คือ
กระจายไปทั่วองค์ประกอบ ถ้ามันขึ้นอยู่กับองค์ประกอบที่คุณคาดหวัง
ช่องว่าง – องค์ประกอบ – ช่องว่าง – องค์ประกอบ – ช่องว่าง
อี
25%
ฉ 75%
ค 33%
ง 66%
ก 50%
ข 50%
รูปที่ 5.17 ตัวอย่างเดียวกับในรูป 5.16 แต่ใช้ flexDirection
ตั้งค่าเป็น 'แถว' ตอนนี้รายการใช้พื้นที่ในแนวนอนภายในแถวค่อนข้าง
กว่าแนวตั้งภายในคอลัมน์
flexContainer เป็นพาเรนต์
คอนเทนเนอร์ของแต่ละตัวอย่าง
ทำให้ลูกๆ
วางในแนวนอน
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 66

แต่ flexbox จะจัดสรรพื้นที่เท่ากันในแต่ละด้านขององค์ประกอบแทน
ให้ผล
ช่องว่าง – องค์ประกอบ – ช่องว่าง – ช่องว่าง – องค์ประกอบ – ช่องว่าง
ในทั้งสองกรณี จำนวนช่องว่างจะเท่ากัน แต่ในระยะหลัง พื้นที่
ระหว่างองค์ประกอบนั้นยิ่งใหญ่กว่า
▪ ช่องว่างระหว่างไม่ใช้การเว้นวรรคที่จุดเริ่มต้นหรือจุดสิ้นสุดของคอนเทนเนอร์ NS
ช่องว่างระหว่างสององค์ประกอบที่ต่อเนื่องกันจะเหมือนกับช่องว่างระหว่าง any
อีกสององค์ประกอบต่อเนื่องกัน
รูปที่ 5.18 แสดงให้เห็นว่าแต่ละคุณสมบัติjustifyContentกระจายอย่างไร
ช่องว่างระหว่างและรอบองค์ประกอบดิ้น ทุกตัวอย่างใช้สององค์ประกอบเพื่อช่วย
พรรณนาถึงสิ่งที่เกิดขึ้น

รายการ 5.7 แสดงรหัสที่ใช้สร้างรูปที่ 5.18 ดูมันอย่างระมัดระวังเพื่อให้-
ยืนหยัดในวิธีการทำงาน แล้วลองทำสิ่งต่อไปนี้: เพิ่มองค์ประกอบเพิ่มเติมในแต่ละข้อสอบ-
เพื่อดูว่าเกิดอะไรขึ้นเมื่อจำนวนรายการเพิ่มขึ้น และตั้งค่าflexDirectionเป็นrow
เพื่อดูว่าเกิดอะไรขึ้นเมื่อวางรายการในแนวนอนแทนที่จะเป็นแนวตั้ง

```
...
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<FlexContainer style={[{justifyContent: 'center'}]}>
<ตัวอย่าง>ศูนย์</Example>
<ตัวอย่าง>ศูนย์</Example>
</FlexContainer>
<FlexContainer style={[{justifyContent: 'flex-start'}]}>
<Example>flex-start</Example>
<Example>flex-start</Example>
</FlexContainer>
<FlexContainer style={[{justifyContent: 'flex-end'}]}>
<ตัวอย่าง>flex-end</Example>
<ตัวอย่าง>flex-end</Example>
</FlexContainer>
<สไตล์ FlexContainer={[{justifyContent: 'space-around'}]}>
<ตัวอย่าง>ช่องว่างรอบๆ</Example>
<ตัวอย่าง>ช่องว่างรอบๆ</Example>
</FlexContainer>
<สไตล์ FlexContainer={[{justifyContent: 'space-between'}]}>
<ตัวอย่าง>ช่องว่างระหว่าง</Example>
<ตัวอย่าง>ช่องว่างระหว่าง</Example>
</FlexContainer>
</ดู>
);
}
...
```

ใช้justifyContent: 'center' option
ใช้justifyContent: 'flex-start' ตัวเลือก
ใช้justifyContent: 'flex-end' option
ใช้justifyContent: 'space-around' option
ใช้justifyContent:
'ช่องว่างระหว่าง' ตัวเลือก
ค้นหาHybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

139

ใช้ exbox เพื่อจัดวางส่วนประกอบ

5.3.4 การจัดแนวเด็กในภาชนะด้วย alignItems

alignItems กำหนดวิธีการจัดแนวเด็กตามแกนรองของคอนเทนเนอร์
คุณสมบัตินี้ถูกประกาศในมุมมองพาเรนต์และมีผลกับลูก flex ของมันเช่นเดียวกับflex-
ทิศทางได้ มีสี่ค่าที่เป็นไปได้สำหรับ alignItems : stretch , center ,
เฟล็กซ์เริ่มต้นและดิ้นสิ้น
การยืดเป็นค่าเริ่มต้น ซึ่งใช้ในรูปที่ 5.17 และ 5.18 ส่วนประกอบแต่ละตัวอย่างคือ
ยืดเพื่อเติมภาชนะหลัก รูปที่ 5.19 ทบทวนรูปที่ 5.16 และแสดงอะไร
ที่เกิดขึ้นกับตัวเลือกอื่น ๆ : ศูนย์ , เฟล็กซ์เริ่มต้นและดิ้นสิ้น เพราะแม่นยำ
ไม่ได้ระบุความกว้างสำหรับส่วนประกอบตัวอย่าง แต่ใช้พื้นที่เพียง hor-
ตามความจำเป็นในการแสดงเนื้อหาแทนที่จะยืดออกเพื่อเติมเต็มพื้นที่
ในกรณีแรกalignItemsถูกตั้งค่าเป็น'ศูนย์' ในกรณีที่สองalignItemsถูกตั้งค่าเป็น
'เริ่มต้นแบบยืดหยุ่น' . และสุดท้าย alignItems ถูกตั้งค่าให้'ดิ้นสิ้น' ใช้รายการ 5.8 เพื่อเปลี่ยน
การจัดตำแหน่งในแต่ละตัวอย่างจากรายการ 5.5

```
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<ดูสไตล์={[styles.flexContainer,
{alignItems: 'center'}]}>
<Example style={[styles.darkgrey]}>50%</Example>
<ตัวอย่าง>ข 50%</Example>
```

เปลี่ยน alignItems

ทรัพย์สินสู่ศูนย์

ศูนย์กลาง

ศูนย์กลาง

flex-start
flex-start
flex-end
flex-end

พื้นที่รอบ ๆ

พื้นที่รอบ ๆ

ช่องว่างระหว่าง

ช่องว่างระหว่าง

รูปที่ 5.18 ตัวอย่างว่า justifyContent ส่งผลต่อการกระจายพื้นที่ระหว่างลูกที่ยืดหยุ่นอย่างไร
องค์ประกอบสำหรับแต่ละตัวเลือกที่รองรับ: center, flex-start, flex-end, space-around และ

ช่องว่างระหว่าง.
ก 50%
ข 50%
ค 33%
ง 66%
อี 25%
ฉ 75%
รูปที่ 5.19 ตัวอย่างที่ดัดแปลงจากรูปที่ 5.16 โดยใช้ค่าที่ไม่ใช่ค่าเริ่มต้น
alignItems คุณสมบัติ: center, flex-start และ flex-end
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 68


```
</ดู>
<ดูสไตล์={[styles.flexContainer,
{alignItems: 'flex-start'}]}>
<Example style={[styles.darkgrey]}>C 33%</Example>
<รูปแบบตัวอย่าง={{flex: 2}}>D 66%</Example>
</ดู>
<ดูสไตล์={[styles.flexContainer,
{alignItems: 'flex-end'}]}>
<Example style={[styles.darkgrey]}>E 25%</Example>
<รูปแบบตัวอย่าง={{flex: 3}}>F 75%</Example>
</ดู>
</ดู>
);
}
```

ตอนนี้คุณได้เห็นวิธีการใช้คุณสมบัติalignItemsอื่นๆและผลกระทบต่อ
เค้าโครงคอลัมน์เริ่มต้น ทำไมคุณไม่ตั้งค่าflexDirectionเป็น'แถว'และดูว่าอะไร
เกิดขึ้น?

5.3.5 การแทนที่การจัดตำแหน่งของคอนเทนเนอร์หลักด้วย alignSelf

จนถึงตอนนี้ คุณสมบัติ flex ทั้งหมดได้ถูกนำไปใช้กับ con-
เทนเนอร์ alignSelfถูกนำไปใช้กับลูกดิ้นแต่ละคนโดยตรง
ด้วยalignSelfคุณสามารถเข้าถึงคุณสมบัติalignItemsสำหรับ
แต่ละองค์ประกอบภายในคอนเทนเนอร์ โดยพื้นฐานแล้วalignSelf
ช่วยให้คุณสามารถแทนที่การจัดตำแหน่งใด ๆ ที่ตั้งค่าไว้บน
คอนเทนเนอร์หลัก ดังนั้นวัตถุลูกสามารถจัดตำแหน่งโดยอิสระจาก

เพื่อนของมัน ตัวเลือกที่ใช้ได้คือauto , stretch , center , flex-
เริ่มต้นและดิ้นสิ้น ค่าเริ่มต้นคือ auto ซึ่งรับ
ค่าจากการตั้งค่าalignItemsของคอนเทนเนอร์หลัก ที่เหลือ-
ing คุณสมบัติส่งผลกระทบต่อเค้าโครงในลักษณะเดียวกับที่สอดคล้อง-
ไอเอ็นจีคุณสมบัติในการalignItems
ในรูป 5.20 คอนเทนเนอร์หลักไม่ได้ตั้งค่าalignItems
ดังนั้นจึงเริ่มต้นที่ยืด ในตัวอย่างแรกค่าอัตโนมัติจะรับ-
มันยืดจากภาชนะที่แม่ของมัน สี่ตัวอย่างถัดไป วาง
ออกมาตรงตามที่คุณคาดหวัง ตัวอย่างสุดท้ายไม่มีalignSelf
ชุดคุณสมบัติ ดังนั้นจึงมีค่าเริ่มต้นเป็นอัตโนมัติและจัดวางเหมือนกับชุดแรก
ตัวอย่าง.
รายการ 5.9 ทำบางสิ่งที่แตกต่างออกไปเล็กน้อย แทนที่จะจัดหาสไตล์ให้กับ .โดยตรง
ตัวอย่างองค์ประกอบคุณสร้างคุณสมบัติองค์ประกอบใหม่: จัด ส่งต่อไปยัง
ตัวอย่างองค์ประกอบและใช้ในการตั้งalignSelf มิฉะนั้น ตัวอย่างจะเหมือนกับ
อื่นๆ อีกมากมายในบทนี้ โดยจะสำรวจผลกระทบของแต่ละค่าที่ใช้กับสไตล์

นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { StyleSheet, Text, View} จาก 'react-native';
เปลี่ยน alignItems เป็น flex-start
เปลี่ยน alignItems เป็น flex-end
ยืด
รถยนต์
ศูนย์กลาง
flex-start
flex-end
ค่าเริ่มต้น
รูป 5.20 How
แต่ละ alignSelf
ทรัพย์สินได้รับผลกระทบ
เลย์เอาต์เมื่อ
คอนเทนเนอร์หลัก
จัดรายการ
ทรัพย์สินถูกตั้งค่าเป็น
ค่าเริ่มต้น
ของยืด
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

141

ใช้ exbox เพื่อจัดวางส่วนประกอบ

```
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<FlexContainer style={[]}>
<Example align='auto'>อัตโนมัติ</Example>
<Example align='stretch'>ยืด</Example>
<Example align='center'>center</Example>
<Example align='flex-start'>flex-start</Example>
<Example align='flex-end'>flex-end</Example>
<ตัวอย่าง>ค่าเริ่มต้น</Example>
</FlexContainer>
</ดู>
);
}
}
const FlexContainer = (อุปกรณ์ประกอบฉาก) => (
<ดู style={[styles.flexContainer,props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</ดู>
);
const ตัวอย่าง = (อุปกรณ์ประกอบฉาก) => (
<ดูสไตล์={[styles.example,
สไตล์ lightgrey,
{alignSelf: props.align || 'อัตโนมัติ'},
props.style
]}>
<ข้อความ>
{อุปกรณ์ประกอบฉาก.เด็ก}
</Text>
</ดู>
);
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ขอบบน: 50,
alignItems: 'ศูนย์',
ดิ้น: 1
},
flexContainer: {
backgroundColor: '#ededed',
ความกว้าง: 120,
```

ความสูง: 180,
ขอบกว้าง: 1,
ระยะขอบ: 10
},
ตัวอย่าง: {
ความสูง: 25,
ระยะขอบด้านล่าง: 5,
พื้นหลังสี: '#666666'
},
});
ตั้งค่า alignSelf เป็น auto,
ซึ่งหยิบ
คอนเทนเนอร์หลัก
ค่าการยืดตัว
ชุดจัดตำแหน่งตัวเอง
อย่างชัดเจนถึง
ยืด
ชุดจัดตำแหน่งตัวเอง
ไปที่ศูนย์
ตั้งค่า alignSelf เป็น flex-start
ตั้งค่า alignSelf เป็น flex-end
ค่าเริ่มต้นสำหรับ alignSelf เป็นค่าอัตโนมัติ
ใช้การจัดตำแหน่ง
คุณสมบัติที่จะตั้งค่า
ตัวอย่างส่วนประกอบ
จัดแนวรายการสไตล์
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 70

5.3.6 การป้องกันรายการที่ถูกตัดด้วย flexWrap

คุณได้เรียนรู้ก่อนหน้านี้ในส่วนนี้ว่าคุณสมบัติflexDirectionรับสองค่า:
คอลัมน์(ค่าเริ่มต้น) และแถว. คอลัมน์จัดวางรายการในแนวตั้ง และแถวจัดวางรายการ
แนวนอน สิ่งที่คุณไม่เคยเห็นคือสถานการณ์ที่รายการไหลออกจากหน้าจอ

เพราะพวกเขาไม่พอดี
flexWrap จะใช้เวลาสองค่า: nowrap และห่อ ค่าเริ่มต้นคือ nowrap ความหมาย
รายการจะไหลออกจากหน้าจอหากไม่พอดี รายการถูกตัดและผู้ใช้ไม่สามารถ
เห็นพวกเขา เมื่อต้องการแก้ไขปัญหานี้ ใช้ค่าตัด
ในรูปที่ 5.21 ตัวอย่างแรกใช้nowrapและสี่เหลี่ยมจะไหลออกจากหน้าจอ NS
แถวของสี่เหลี่ยมถูกตัดออกที่ขอบด้านขวา ตัวอย่างที่สองใช้wrapและ the
สี่เหลี่ยมล้อมรอบและเริ่มแถวใหม่ รายการ 5.10 แสดงรหัส

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า { StyleSheet, Text, View} จาก 'react-native';
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์<{}> {
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<NoWrapContainer>
<ตัวอย่าง>นาวแร็ป</Example>
นาวแรป
ห่อบี
1
1
2
3
4
2
3
```

รูปที่ 5.21 ตัวอย่างของสองภาชนะที่ล้น:
อันหนึ่งที่ตั้งค่า flexWrap เป็น nowrap และอีกอันด้วย
flexWrap ตั้งค่าเป็น wrap
flexWrap ถูกตั้งค่าเป็น nowrap: the
สี่เหลี่ยมล้นออกจากหน้าจอ
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 71

143
ใช้ exbox เพื่อจัดวางส่วนประกอบ

```
<ตัวอย่าง>1</Example>
<ตัวอย่าง>2</Example>
<ตัวอย่าง>3</Example>
<ตัวอย่าง>4</Example>
```

```
</NoWrapContainer>
<ห่อคอนเทนเนอร์>
<ตัวอย่าง>ห่อ B</Example>
<ตัวอย่าง>1</Example>
<ตัวอย่าง>2</Example>
<ตัวอย่าง>3</Example>
<ตัวอย่าง>4</Example>
</WrapContainer>
</ดู>
);
}
}
const NoWrapContainer = (อุปกรณ์ประกอบฉาก) => (
<ดู style={[styles.noWrapContainer,props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</ดู>
);
const WrapContainer = (อุปกรณ์ประกอบฉาก) => (
<ดู style={[styles.wrapContainer,props.style]}>
{อุปกรณ์ประกอบฉาก.เด็ก}
</ดู>
);
const ตัวอย่าง = (อุปกรณ์ประกอบฉาก) => (
<ดู style={[styles.example,props.style]}>
<ข้อความ>
{อุปกรณ์ประกอบฉาก.เด็ก}
</Text>
</ดู>
);
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ขอบบน: 150,
ดื้น: 1
},
noWrapContainer: {
backgroundColor: '#ededed',
flexDirection: 'แถว',
flexWrap: 'โนแร็ป',
ขอบกว้าง: 1,
ระยะขอบ: 10
},
คอนเทนเนอร์ห่อ: {
backgroundColor: '#ededed',
flexDirection: 'แถว',
flexWrap: 'ห่อ'
```

ขอบกว้าง: 1,
ระยะขอบ: 10
},
flexWrap ถูกตั้งค่าให้ห่อ:
แถวของสี่เหลี่ยมห่อ
เพื่อเริ่มบรรทัดใหม่
ใช้ noWrapContainer
สไตล์สำหรับตัวอย่างแรก
ใช้ wrapContainer
สไตล์สำหรับตัวอย่างที่สอง
ตั้งค่า flexDirection เป็นแถว
และ flexWrap เป็น nowrap
ตั้งค่า flexDirection เป็นแถว
และ flexWrap เพื่อห่อ
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 72

ตัวอย่าง: {
ความกว้าง: 100,
ความสูง: 100,
ระยะขอบ: 5,
พื้นหลังสี: '#666666'
},
});
ง่ายต่อการดูว่าพฤติกรรมใดดีกว่าเมื่อวางกระเบื้อง แต่คุณอาจมา
ในสถานการณ์ที่nowrapจะให้บริการคุณได้ดียิ่งขึ้น ไม่ว่าจะด้วยวิธีใด ตอนนี้คุณควร
มีความเข้าใจที่ชัดเจนเกี่ยวกับ flexbox และวิธีต่างๆ ที่จะช่วยให้คุณสร้างการตอบสนอง
เลย์เอาต์แบบซีฟใน React Native

สรุป

- เมื่อปรับขนาดรายการสำหรับการแสดงผล iOS จะใช้จุดและ Android ใช้ความหนาแน่นแบบไม่จำกัด
พิกเซลจี้ ระบบการวัดต่างกันแต่ควรมีน้อย
ผลกระทบต่อการพัฒนาเว้นแต่คุณต้องการกราฟิกที่สมบูรณ์แบบพิกเซล
- บางสไตล์มีเฉพาะในแพลตฟอร์มเดียวเท่านั้น ShadowPropTypeIOS คือ

ใช้ได้เฉพาะบน iOS และรู้จักการยกระดับบน Android เท่านั้น
- ส่วนประกอบสามารถย้ายในทิศทาง x และ y โดยใช้translateX และ
แปลY แปลง
- ส่วนประกอบสามารถหมุนรอบแกน x-, y- และ z ได้โดยใช้การหมุนX , หมุนY ,
และrotateZ จุดหมุนคือตำแหน่งเดิมของวัตถุมาก่อน
มีการใช้การแปลงใดๆ
- ส่วนประกอบสามารถปรับขนาดได้ในทิศทาง x และ y เพื่อสร้างส่วนประกอบ
เติบโตหรือหดตัว
- ส่วนประกอบสามารถเอียงในทิศทาง x และ y ได้เช่นกัน
- สามารถใช้การแปลงหลายแบบพร้อมกันได้ แต่ลำดับที่
เป็นเรื่องเฉพาะ การหมุนส่วนประกอบจะเปลี่ยนทิศทางของ
องค์ประกอบสำหรับการแปลงในภายหลัง
- flexDirection คุณสมบัติกำหนดแกนหลักเป็นค่าเริ่มต้นคอลัมน์
(แกน y).
- justifyContent กำหนดคุณสมบัติว่ารายการที่ควรจะออกมาวางตามแนว
แกนหลัก
- alignItems กำหนดคุณสมบัติว่ารายการที่ควรจะออกมาวางตามแนวที่สอง
แกนอารี
- alignSelf คุณสมบัติสามารถใช้ในการแทนที่alignItems คุณสมบัติ speci-
fied โดยคอนเทนเนอร์หลัก
- flexWrap คุณสมบัติบอก flexbox วิธีการจัดการกับรายการที่จะมักจะ
ล้นออกจากหน้าจอ
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

145

การนำทาง

บทนี้ครอบคลุม

- การนำทางใน React Native กับเว็บ
- การนำทางโดยใช้แท็บ กอง และลิ้นชัก
- การจัดการเนวิเกเตอร์ที่ซ้อนกัน
- ส่งข้อมูลและวิธีการระหว่างเส้นทาง

หนึ่งในส่วนสำคัญของการทำงานในแอปพลิเคชันมือถือคือการนำทาง
ก่อนสร้างแอปพลิเคชัน ฉันแนะนำให้คุณใช้เวลาในการวางแผน
วิธีที่คุณต้องการให้แอปจัดการการนำทางและการกำหนดเส้นทาง บทนี้ครอบคลุมถึง
การนำทางหลักสามประเภทโดยทั่วไปสำหรับแอปพลิเคชันมือถือ: แบบแท็บ แบบสแต็ก
และการนำทางแบบลิ้นชัก
การนำทางแบบแท็บมักมีแท็บที่ด้านบนหรือด้านล่างของหน้าจอ
การกดแท็บจะนำคุณไปยังหน้าจอที่สัมพันธ์กับแท็บนั้น แอพยอดนิยมมากมาย
เช่น Twitter, Instagram และ Facebook ใช้การนำทางประเภทนี้ใน
หน้าจอหลัก
การนำทางแบบกองซ้อนเปลี่ยนจากหน้าจอหนึ่งไปอีกหน้าจอหนึ่ง แทนที่เคอร์-
เช่าหน้าจอ และมักจะใช้การเปลี่ยนภาพเคลื่อนไหวบางประเภท จากนั้นคุณสามารถ
ถอยหลังหรือเดินหน้าต่อไปในกอง คุณสามารถนึกถึง stack-based
การนำทางเหมือนอาร์เรย์ของส่วนประกอบ: การผลักส่วนประกอบใหม่เข้าไปในอาร์เรย์
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

นำคุณไปยังหน้าจอของส่วนประกอบใหม่ หากต้องการย้อนกลับ คุณต้องเปิดหน้าจอสุดท้ายจาก
สแต็คและถูกนำทางไปยังหน้าจอก่อนหน้า ไลบรารีการนำทางส่วนใหญ่จัดการสิ่งนี้
popping และผลักดันสำหรับคุณ
การนำทางแบบลิ้นชักมักจะเป็นเมนูด้านข้างที่โผล่ออกมาจากด้านซ้าย
หรือด้านขวาของหน้าจอและแสดงรายการตัวเลือก เมื่อคุณกดตัวเลือก
ลิ้นชักปิดและคุณจะถูกนำไปที่หน้าจอใหม่
กรอบงาน React Native ไม่รวมไลบรารีการนำทาง เมื่อสร้าง
การนำทางในแอป React Native คุณต้องใช้ไลบรารีการนำทางของบุคคลที่สาม NS
มีไลบรารีการนำทางที่ดีไม่กี่แห่ง แต่ในบทนี้ ฉันใช้ React Navigation

เป็นไลบรารีการนำทางที่เลือกเพื่อสร้างแอปสาธิต การนำทางตอบสนอง
ห้องสมุดได้รับการแนะนำโดยทีมงาน React Native และดูแลโดยคนจำนวนมากใน
ชุมชน React และ React Native
React Navigation คือการนำระบบนำทางแบบ JavaScript ไปใช้ การเปลี่ยนแปลงทั้งหมด
และการควบคุมถูกจัดการโดย JavaScript บางทีมชอบโซลูชันดั้งเดิมสำหรับหลาย ๆ คน
เหตุผล: ตัวอย่างเช่น พวกเขาอาจเพิ่ม React Native ให้กับแอพเนทีฟที่มีอยู่และ
ต้องการให้การนำทางมีความสอดคล้องกันตลอดทั้งแอป หากคุณสนใจในความเป็นพื้นเมือง
โซลูชันการนำทาง ลองดู React Native Navigation ซึ่งเป็นโอเพ่นซอร์ส React Native
ไลบรารีการนำทางที่สร้างและดูแลโดยวิศวกรของ Wix

6.1 ตอบสนองการนำทางดั้งเดิมกับการนำทางเว็บ

เนื่องจากกระบวนทัศน์การนำทางบนเว็บนั้นแตกต่างจาก React . มาก
Native การนำทางเป็นสิ่งกีดขวางสำหรับนักพัฒนาหลายคนที่เพิ่งเริ่มใช้ React Native บน
ทางเว็บเราเคยชินกับการทำงานกับ URL มีหลายวิธีในการนำทางไปยังใหม่
เส้นทาง ขึ้นอยู่กับเฟรมเวิร์กหรือสภาพแวดล้อม แต่โดยทั่วไปคุณต้องการส่ง
ผู้ใช้ไปยัง URL ใหม่และอาจเพิ่มพารามิเตอร์ URL บางอย่างหากจำเป็น
ใน React Native เส้นทางจะขึ้นอยู่กับส่วนประกอบ คุณโหลดหรือแสดงคอมโพ-
nent โดยใช้เนวิเกเตอร์ที่คุณกำลังทำงานด้วย ขึ้นอยู่กับว่าเป็นแท็บตาม
ตามสแตก ตามลิ้นชัก หรือรวมกันเหล่านี้ การกำหนดเส้นทางก็จะแตกต่างกันด้วย ดี
อธิบายทั้งหมดนี้เมื่อคุณสร้างแอปสาธิตในส่วนถัดไป
คุณต้องติดตามข้อมูลและสถานะตลอดเส้นทางและอาจ
วิธีการเข้าถึงที่กำหนดไว้ที่อื่นในแอป ดังนั้นจึงมีกลยุทธ์เกี่ยวกับข้อมูลและวิธีการ
การแบ่งปันเป็นสิ่งสำคัญ คุณสามารถจัดการข้อมูลและวิธีการที่ระดับบนสุด โดยที่
การนำทางถูกกำหนดหรือใช้ไลบรารีการจัดการสถานะเช่น Redux หรือ MobX ใน
ตัวอย่างเช่น คุณจะจัดการข้อมูลและวิธีการในชั้นเรียนที่ระดับบนสุดของแอป

6.2 การสร้างแอพที่ใช้การนำทาง

ในบทนี้ คุณจะได้เรียนรู้วิธีใช้งานการนำทางโดยการสร้างแอปที่
ใช้การนำทางทั้งแบบแท็บและแบบสแต็ก แอปที่คุณจะสร้างเรียกว่าเมือง
มันแสดงในรูปที่ 6.1 เป็นแอปท่องเที่ยวที่ให้คุณตามทันทุกเมืองที่คุณไป
หรือต้องการเยี่ยมชม คุณสามารถเพิ่มสถานที่ในแต่ละเมืองที่คุณต้องการเยี่ยมชมได้
การนำทางหลักเป็นแบบแท็บ และหนึ่งในแท็บมีการนำทางแบบสแต็ก
ชั่น แท็บด้านซ้ายแสดงรายการเมืองที่คุณสร้าง และแท็บด้านขวาประกอบด้วยฟอร์ม
เพื่อสร้างเมืองใหม่ ที่แท็บด้านซ้าย คุณสามารถกดแต่ละเมืองเพื่อดูเมืองได้เช่นกัน
ดูและสร้างสถานที่ในเมือง

หน้า 75

147

การสร้างแอพที่ใช้การนำทาง

ในการเริ่มต้น ให้สร้างแอปพลิเคชัน React Native ใหม่ ในเทอร์มินัลของคุณ ให้ไปที่ an
ไดเร็กทอรีว่าง และติดตั้งแอปพลิเคชัน React Native ใหม่โดยใช้ React Native CLI:

```
react-native init CitiesApp
```

ถัดไป นำทางไปยังไดเร็กทอรีใหม่ และติดตั้งการขึ้นต่อกันสองรายการ: React Navigation
และ uuid React Navigation คือไลบรารีการนำทาง และ uuid จะถูกใช้เพื่อสร้าง
รหัสเฉพาะสำหรับเมืองต่างๆ เพื่อระบุไม่ซ้ำกัน:

```
cd CitiesApp
npm ติดตั้ง react-navigation uuid
```

มาเริ่มสร้างส่วนประกอบกันเถอะ! สร้างไดเร็กทอรีหลักใหม่ที่เรียกว่า src in
รูทของแอปพลิเคชัน ไดเร็กทอรีนี้จะเก็บโค้ดใหม่เกือบทั้งหมดสำหรับแอป
ในไดเร็กทอรีใหม่นี้ เพิ่มไดเร็กทอรีย่อยหลักสามไดเร็กทอรี: Cities, AddCity และส่วนประกอบ
เนื่องจากการนำทางหลักเป็นแบบแท็บ คุณจะต้องแยกแอปพลิเคชันหลักออกเป็น
สององค์ประกอบหลัก (CitiesและAddCity) แต่ละองค์ประกอบมีแท็บของตัวเอง The AddCity
โฟลเดอร์จะมีเพียงองค์ประกอบเดียวคือ AddCity.js โฟลเดอร์ Cities จะมี
สององค์ประกอบ: Cities.js เพื่อดูรายชื่อเมือง และ City.js เพื่อดูแต่ละเมือง
โฟลเดอร์ส่วนประกอบจะเก็บส่วนประกอบที่นำกลับมาใช้ใหม่ได้ ในกรณีนี้จะถือ
องค์ประกอบเดียว

คุณจะมีไฟล์ src/index.js และ src/theme.js src/index.js จะเก็บ nav-
การกำหนดค่า igation และ theme.js จะเป็นที่ที่คุณเก็บการกำหนดค่าตามธีมได้—
ในกรณีนี้คือการกำหนดค่าสีหลัก รูปที่ 6.2 แสดงความสมบูรณ์ของโครงการ
โครงสร้างโฟลเดอร์

ตอนนี้คุณได้สร้างโครงสร้างโฟลเดอร์และติดตั้งการพึ่งพาที่จำเป็นแล้ว
cies มาเขียนโค้ดกัน ไฟล์แรกที่คุณจะใช้งานคือ src/theme.js ที่นี่ คุณจะ
ตั้งค่าสีหลักและทำให้ส่งออกเพื่อใช้ในแอพได้ สีของธีมที่ฉันมีอยู่
เลือกสำหรับแอพเป็นสีน้ำเงิน แต่คุณสามารถใช้สีใดก็ได้ที่คุณต้องการ แอพจะทำงาน
เหมือนกันถ้าคุณเปลี่ยนค่าสีในไฟล์นี้

รูปที่ 6.1 แอปเมืองที่เสร็จสมบูรณ์พร้อมหน้าจอสำหรับเพิ่มเมือง รายชื่อเมือง การดูรายละเอียดเมือง และ
ดูสถานที่ภายในเมือง

หน้า 76


```
const สี = {
หลัก: '#1976D2'
}
ส่งออก {
สี
}
```

คุณสามารถนำเข้าสีหลักนี้ทั่วทั้งแอปพลิเคชันหากต้องการและเปลี่ยน
ไว้ในที่เดียวหากคุณเลือกที่จะทำเช่นนั้น
ถัดไป แก้ไข src/index.js เพื่อสร้างการกำหนดค่าการนำทางหลัก คุณจะสร้าง
ทั้งสองอินสแตนซ์การนำทางที่นี่: การนำทางแบบแท็บและการนำทางแบบสแต็ก

```
นำเข้าปฏิกิริยาจาก 'ปฏิกิริยา'
นำเข้าเมืองจาก './Cities/Cities'
นำเข้าเมืองจาก './Cities/City'
นำเข้า AddCity จาก './AddCity/AddCity'
นำเข้า { สี } จาก './theme'
นำเข้า { createBottomTabNavigator,
createStackNavigator } จาก 'react-navigation'
ตัวเลือก const = {
การนำทางตัวเลือก: {
สไตล์ส่วนหัว: {
backgroundColor: colors.primary
},
```

รูปที่ 6.2 โครงสร้างโฟลเดอร์ src ที่สมบูรณ์

นำเข้าส่วนประกอบทั้งสาม
ให้อยู่ในขอบเขตของไฟล์
นำเข้าสีจากธีม
นำเข้าทั้งสอง
ตัวนำทางในการเข้าถึง
จากการนำทางตอบสนอง
สร้างอ็อบเจ็กต์ตัวเลือกที่จะถือ
การกำหนดค่าสำหรับตัวนำทางสแต็ก

หน้า 77

149

การสร้างแอพที่ใช้การนำทาง

ส่วนหัวTintColor: '#fff'

}

}

const CitiesNav = createStackNavigator ({

เมือง: { หน้าจอ: เมือง },

เมือง: { หน้าจอ: เมือง }

}, ตัวเลือก)

แท็บ const = createBottomTabNavigator ({

เมือง: { หน้าจอ: CitiesNav },

AddCity: { หน้าจอ: AddCity }

})

ส่งออกแท็บเริ่มต้น

เมื่อคุณสร้างอ็อบเจ็กต์optionsสแต็กเนวิเกเตอร์จะวางส่วนหัวโดยอัตโนมัติ
ที่ด้านบนสุดของแต่ละเส้นทาง ส่วนหัวมักจะเป็นที่ที่คุณจะมีชื่อเรื่องของปัจจุบัน
เส้นทางเช่นเดียวกับปุ่มเช่นปุ่มย้อนกลับ ตัวเลือกวัตถุนอกจากนี้ยังกำหนด Back-
สีพื้นและสีอ่อนของส่วนหัว
สำหรับอินสแตนซ์การนำทางแรกcreateStackNavigatorรับสองอาร์กิวเมนต์: the
การกำหนดค่าเส้นทางและการกำหนดค่าใดๆ เกี่ยวกับสิ่งต่างๆ เช่น การจัดรูปแบบเพื่อนำไปใช้กับ
การนำทาง คุณส่งผ่านสองเส้นทางเป็นอาร์กิวเมนต์แรก และอ็อบเจ็กต์ตัวเลือกเป็น
อาร์กิวเมนต์ที่สอง
ถัดไป อัปเดต App.js เพื่อรวมการนำทางใหม่และแสดงผลเป็นรายการหลัก
จุด. นอกเหนือจากการแสดงองค์ประกอบการนำทางแล้ว App.js จะมีและ
โทรลวิธีการและข้อมูลใด ๆ ที่จะจัดให้มีขึ้นในแอปพลิเคชัน

นำเข้า React { ส่วนประกอบ } จาก 'react';

นำเข้า {

แพลตฟอร์ม,

สไตล์ชีต,

ข้อความ,

ดู

} จาก 'react-native';

นำเข้าแท็บจาก './src'

```
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์ {
รัฐ = {
เมือง: []
}
addCity = (เมือง) => {
เมือง const = this.state.cities
city.push(เมือง)
this.setState({ เมือง })
}
addLocation = (ที่ตั้ง, เมือง) => {
ดัชนี const = this.state.cities.findIndex (รายการ => {
ส่งคืน item.id === city.id
})
```

สร้างอินสแตนซ์การนำทางแรก
สร้างเนวิเกเตอร์แท็บโดยใช้
ตัวนำทางสแต็ก CitiesNav
สำหรับหนึ่งแท็บและ AddCity
องค์ประกอบสำหรับแท็บที่สอง
นำเข้าการนำทางจาก src/index.js
สร้างสถานะเริ่มต้น
ของเมือง, อาร์เรย์ที่ว่างเปล่า
เพิ่มเมืองใหม่ในรายการที่มีอยู่
ของเมืองที่เก็บไว้ในรัฐ
เพิ่มตำแหน่งให้กับอาร์เรย์
ของสถานที่ในเมืองที่เลือก

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 78


```
const selectedCity = this.state.cities[ดัชนี]
เลือกเมือง.locations.push(ที่ตั้ง)
เมือง const = [
...this.state.cities.slice(0, ดัชนี),
เลือกเมือง
...this.state.cities.slice(ดัชนี + 1)
]
```

```
this.setState({
เมือง
})
}
เรนเดอร์ () {
กลับ (
<แท็บ
หน้าจออุปกรณ์ประกอบฉาก={{
เมือง: this.state.cities,
addCity: this.addCity,
addLocation: this.addLocation
}}
/>
)
}
}
```

App.js มีฟังก์ชันหลักสามส่วน มันสร้างสถานะเริ่มต้นของแอพ: an
ว่างอาร์เรย์เรียกว่าเมือง แต่ละเมืองจะเป็นวัตถุและจะมีชื่อ, ประเทศ,
ID และอาร์เรย์ของสถานที่ addCityวิธีช่วยให้คุณสามารถเพิ่มเมืองใหม่ในเมือง
อาร์เรย์ที่เก็บไว้ในสถานะ addLocationวิธีการระบุเมืองที่คุณต้องการเพิ่ม
ตำแหน่งเพื่ออัปเดตเมืองและรีเซ็ตสถานะด้วยข้อมูลใหม่
React Navigation มีวิธีการส่งผ่านวิธีการเหล่านี้และสถานะลงไปยังทุกเส้นทาง
ถูกใช้โดยเครื่องนำทาง เมื่อต้องการทำสิ่งนี้ ให้ส่งพร็อพที่เรียกว่าscreenPropsประกอบด้วย
สิ่งที่คุณต้องการเข้าถึง จากนั้น จากภายในเส้นทางใดthis.props.screenProps
ให้การเข้าถึงข้อมูลหรือวิธีการ
ถัดไป คุณจะต้องสร้างส่วนประกอบที่ใช้ซ้ำได้ที่เรียกว่าCenterMessageซึ่งใช้ใน
Cities.js และ City.js (src/components/CenterMessage.js) จะแสดงข้อความเมื่อ
อาร์เรย์ว่างเปล่า ตัวอย่างเช่น เมื่อแอปเริ่มต้นครั้งแรก จะไม่มีรายชื่อเมืองใดๆ
คุณสามารถแสดงข้อความดังแสดงในรูปที่ 6.3 แทนที่จะแสดงเพียงหน้าจอเปล่า

```
นำเข้าปฏิกิริยาจาก 'ปฏิกิริยา'
นำเข้า {
ข้อความ,
ดู,
สไตล์ชีต
} จาก 'react-native'
นำเข้า { สี } จาก '../theme'
const CenterMessage = ({ ข้อความ }) => (
<ดู style={styles.emptyContainer}>
```

ส่งกลับส่วนประกอบ Tabs และผ่าน

ในวัตถุ screenProps ที่มี
เมืองอาร์เรย์ วิธีการ addCity และ
วิธีการ addLocation

หน้า 79

151
การสร้างแอพที่ใช้การนำทาง
<Text style={styles.message}>{message}</Text>
</ดู>
)
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์ว่างเปล่า: {
ช่องว่างภายใน: 10,
borderBottom ความกว้าง: 2,
borderBottomColor: colors.primary
},
ข้อความ: {
alignSelf: 'ศูนย์',
ขนาดตัวอักษร: 20
}
})
ส่งออกค่าเริ่มต้น CenterMessage
องค์ประกอบนี้ตรงไปตรงมา เป็นองค์ประกอบไร้สัญชาติที่ได้รับเพียงข้อความ
ปราชญ์เป็นพร็อพและแสดงข้อความพร้อมกับสไตล์บางอย่าง
ถัดไป ใน src/AddCity/AddCity.js ให้สร้างองค์ประกอบAddCityที่จะช่วยให้คุณ
เพื่อเพิ่มเมืองใหม่ให้กับอาร์เรย์เมือง (ดูรูปที่ 6.4) ส่วนประกอบนี้จะมี a
รูปที่ 6.3 CenterMessage ที่นำกลับมาใช้ใหม่ได้
คอมโพเนนท์แสดงข้อความตรงกลาง
ภายในจอแสดงผล
รูปที่ 6.4 แท็บ AddCity ช่วยให้
ผู้ใช้เพื่อป้อนชื่อเมืองใหม่และ
ชื่อประเทศ.

หน้า 80

รูปแบบที่มีการป้อนข้อความสองแบบ: หนึ่งสำหรับเก็บชื่อเมืองและอีกอันสำหรับเก็บชื่อประเทศ นอกจากนี้ปุ่มจะเรียกaddCityวิธีจาก App.js.

```
นำเข้าปฏิกิริยาจาก 'ปฏิกิริยา'
นำเข้า {
ดู,
ข้อความ,
สไตล์ชีต,
อินพุตข้อความ,
TouchableOpacity
} จาก 'react-native'
นำเข้า uuidV4 จาก 'uuid/v4'
นำเข้า { สี } จาก '../theme'
ส่งออกคลาสเริ่มต้น AddCity ขยาย React.Component {
รัฐ = {
เมือง: '',
ประเทศ: '',
}
onChangeText = (คีย์ ค่า) => {
this.setState({ [คีย์]: ค่า })
}
ส่ง = () => {
if (this.state.city === '' || this.state.country === '') {
alert('กรุณากรอกแบบฟอร์ม')
}
เมือง const = {
เมือง: this.state.city,
ประเทศ: this.state.country,
รหัส: uuidV4(),
สถานที่: []
}
this.props.screenProps.addCity(เมือง)
this.setState({
เมือง: '',
ประเทศ: ''
}, () => {
this.props.navigation.navigate('เมือง')
})
}
เรนเดอร์ () {
กลับ (
```

<ดู style={styles.container}>
<Text style={styles.heading}>เมือง</Text>
<ป้อนข้อความ
ตัวยึด = 'ชื่อเมือง'
onChangeText={val => this.onChangeText('เมือง', val)}
style={styles.input}
ค่า={this.state.city}
รัฐเริ่มต้นถือชื่อเมืองและประเทศ
ชื่อ ทั้งสองตั้งต้นเป็นสตริงว่าง
อัพเดทสถานะด้วยเมืองหรือ
ค่าชื่อ สิ่งนี้จะแนบมากับ
TextInput และจะเริ่มทำงานทุกครั้งที่
การเปลี่ยนแปลงค่าอินพุต
ถือมาก
ของ
ฟังก์ชั่น
สำหรับสิ่งนี้
ส่วนประกอบ
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 81

153
การสร้างแอพที่ใช้การนำทาง
/>
<ป้อนข้อความ
placeholder='ชื่อประเทศ'
onChangeText={val => this.onChangeText('ประเทศ', val)}
style={styles.input}
ค่า={this.state.country}
/>
<TouchableOpacity onPress={this.submit}>
<ดู style={styles.button}>
<Text style={styles.buttonText}>เพิ่มเมือง</Text>
</ดู>
</TouchableOpacity>
</ดู>
)
}

}
ขั้นแรก คุณต้องตรวจสอบว่าทั้งเมืองและประเทศไม่ใช่สตริงว่าง ถ้า
หรือทั้งสองอย่างว่างเปล่า คุณกลับมา เพราะคุณไม่ต้องการเก็บข้อมูลเว้นแต่
กรอกข้อมูลทั้งสองช่อง ถัดไป คุณสร้างวัตถุเพื่อให้เมืองถูกเพิ่มไปยัง
เมืองอาร์เรย์ นำค่าเมืองและประเทศที่มีอยู่ที่เก็บไว้ในรัฐและเพิ่ม
ค่า ID โดยใช้เมธอดuuidV4และอาร์เรย์ตำแหน่งว่าง เรียกสิ่งนี้ว่าอุปกรณ์ประกอบฉาก
screenProps.addCity ผ่านเมืองใหม่ ถัดไป รีเซ็ตสถานะเพื่อล้างใดๆ
ค่าที่เก็บไว้ในสถานะ สุดท้าย นำทางผู้ใช้ไปยังแท็บเมืองเพื่อแสดง
รายชื่อเมืองที่เพิ่มเมืองใหม่โดยเรียกthis.props.navigation.navigationและ
ผ่านในสายของเส้นทางเพื่อนำทางไปในกรณีนี้'เมือง'
ทุกองค์ประกอบที่เป็นหน้าจอในเนวิเกเตอร์จะสามารถเข้าถึงอุปกรณ์ประกอบฉากสองอย่างได้โดยอัตโนมัติ:
screenProps และนำทาง ในรายการ 6.3 เมื่อคุณสร้างองค์ประกอบการนำทาง
ตรวจวัดค่าคุณผ่านในสามscreenProps ในวิธีการส่งคุณเรียกthis.props
screenProps.addCity การเข้าถึงและเรียกใช้เมธอด screenProps นี้ คุณยัง
เข้าถึงเสานำทางโดยการเรียกthis.props.navigation.navigate นำทางคือ
สิ่งที่คุณใช้เพื่อนำทางระหว่างเส้นทางต่างๆ ใน React Navigation
ถัดไป เพิ่มสไตล์สำหรับส่วนประกอบนี้ รหัสนี้อยู่ต่ำกว่าคำจำกัดความของคลาสใน
src/AddCity/AddCity.js

รูปแบบ const = StyleSheet.create ({
ปุ่ม: {
ความสูง: 50,
พื้นหลังสี: '#666',
justifyContent: 'ศูนย์',
alignItems: 'ศูนย์',
ระยะขอบ: 10
},
ปุ่มข้อความ: {
สี: 'ขาว',
ขนาดตัวอักษร: 18
},
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หัวข้อ: {
สี: 'ขาว',
ขนาดตัวอักษร: 40,
ระยะขอบด้านล่าง: 10,
alignSelf: 'ศูนย์'
},
คอนเทนเนอร์: {
backgroundColor: สีหลัก,
ดิ้น: 1,
justifyContent: 'ศูนย์กลาง'
},
ป้อนข้อมูล: {
ระยะขอบ: 10,
พื้นหลังสี: 'สีขาว',
paddingแนวนอน: 8,
ส่วนสูง: 50
}
})
ตอนนี้ สร้าง src/Cities/Cities.js เพื่อแสดงรายการเมืองทั้งหมดที่แอปจัดเก็บและอนุญาตให้ผู้ใช้เพื่อนำทางไปยังแต่ละเมือง (ดูรูปที่ 6.5) ฟังก์ชันจะแสดงใน f-
รายชื่อต่ำและสไตล์อยู่ในรายการ 6.8
รูปที่ 6.5 Cities.js แสดงรายการเมืองที่เคยไปมาแล้ว
เพิ่มไปยังแอปพลิเคชัน
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 83

155
การสร้างแอพที่ใช้การนำทาง

นำเข้าปฏิกิริยาจาก 'ปฏิกิริยา'
นำเข้า {
ดู,
ข้อความ,
สไตล์ชีต,
สัมผัสได้โดยไม่มีข้อเสนอแนะ
ScrollView
} จาก 'react-native'
นำเข้า CenterMessage จาก '../components/CenterMessage'
นำเข้า { สี } จาก '../theme'
ส่งออกคลาสเริ่มต้น Cities ขยาย React.Component {

```
navigationOptions แบบคงที่ = {
ชื่อเรื่อง: 'เมือง',
headerTitleStyle: {
สี: 'ขาว',
ขนาดตัวอักษร: 20,
fontWeight: '400'
}
}
นำทาง = (รายการ) => {
this.props.navigation.navigate('เมือง' { เมือง: รายการ })
}
เรนเดอร์ () {
const { screenProps: { เมือง } } = this.props
กลับ (
<ScrollView contentContainerStyle={[!cities.length && { ยืดหยุ่น: 1 }]}>
<ดูสไตล์={[!cities.length &&
{ justifyContent: 'center', flex: 1 }]}>
{
!cities.length && <CenterMessage message='ไม่มีเมืองที่บันทึกไว้!'/>
}
{
เมือง.map((รายการ, ดัชนี) => (
<สัมผัสได้โดยไม่มีข้อเสนอแนะ
onPress={() => this.navigate(item)} คีย์={index} >
<ดู style={styles.cityContainer}>
<Text style={styles.city}>{item.city}</Text>
<Text style={styles.country}>{item.country}</Text>
</ดู>
</TouchableWithoutFeedback>
))
}
</ดู>
</ScrollView>
)
}
}
```

ในรายการนี้ คุณต้องนำเข้าคอมโพเนนต์CenterMessageก่อน ตอบสนองการนำทางมีวิธีควบคุมตัวเลือกบางอย่างรอบๆ การนำทางภายในเส้นทาง โดยคุณสามารถประกาศคุณสมบัติnavigationOptionsแบบคงที่ในคลาสและประกาศการกำหนดค่า

นำเข้า CenterMessage

องค์ประกอบที่สร้างขึ้นในรายการ6.4

ประกาศ navigationOptions แบบคงที่

ทรัพย์สินในชั้นเรียนและประกาศ

การกำหนดค่าสำหรับเส้นทางนี้

ผ่านไปในเมืองเป็นข้อโต้แย้งที่สอง
ไปที่ this.props.navigation.navigate
การเข้าถึงและ
ทำลาย
เมืองอาร์เรย์จาก
หน้าจออุปกรณ์ประกอบฉาก
พร็อพที่มีอยู่ใน
ส่วนประกอบ
ตรวจสอบว่า
เมืองอาร์เรย์คือ
ว่างเปล่า. ถ้าใช่,
แสดงผู้ใช้
ข้อความว่า
ไม่มี
เมืองในปัจจุบัน
ในแอป
แผนที่ทั่วเมืองใน
อาร์เรย์ที่แสดง
ชื่อเมืองและประเทศ
ชื่อ. ยังแนบ
นำทางไปยัง
สัมผัสได้โดยไม่มีข้อเสนอแนะ
ส่วนประกอบ.
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 84

156
C HAPTER 6 การนำทาง
สำหรับเส้นทาง ในกรณีนี้ คุณต้องการกำหนดหัวเรื่องและกำหนดรูปแบบหัวเรื่อง ดังนั้นให้กำหนดค่า a
คุณสมบัติหัวเรื่องและส่วนหัวTitleStyle
สำรวจวิธีการเรียกthis.props.navigation.navigateและผ่านใน
ชื่อเส้นทางตลอดจนเมืองที่จะเข้าถึงในเส้นทางเมือง ผ่านในเมืองเป็นที่สอง

การโต้แย้ง; ในเส้นทางCityคุณจะสามารถเข้าถึงคุณสมบัตินี้ในprops.navigation state.params วิธีการเรนเดอร์เข้าถึงและทำลายโครงสร้างอาร์เรย์เมือง นอกจากนี้ยัง รวมตรรกะเพื่อตรวจสอบว่าอาร์เรย์เมืองว่างเปล่าหรือไม่ ถ้าใช่ ให้แสดงผู้ใช้ an ข้อความที่เหมาะสม คุณทำแผนที่เหนือเมืองทั้งหมดในอาร์เรย์ โดยแสดงชื่อเมือง และชื่อประเทศ การแนบวิธีการนำทางเข้ากับTouchableWithoutFeedback คอมโพเนนต์ให้ผู้ใช้นำทางไปยังเมืองโดยกดที่ใดก็ได้ในเมือง

รูปแบบ const = StyleSheet.create ({
เมืองคอนเทนเนอร์: {
ช่องว่างภายใน: 10,
borderBottom ความกว้าง: 2,
borderBottomColor: colors.primary
},
เมือง: {
ขนาดตัวอักษร: 20,
},
ประเทศ: {
สี: 'rgba(0, 0, 0, .5)'
},
})
รูปที่ 6.6 City.js แสดงตำแหน่งภายในเมือง
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc facebook.com/somsacki

---

หน้า 85

157
การสร้างแอพที่ใช้การนำทาง
ถัดไป สร้างองค์ประกอบเมือง (src/Cities/City.js) เพื่อเก็บตำแหน่งสำหรับแต่ละ เมืองรวมถึงรูปแบบที่ช่วยให้ผู้ใช้สร้างที่ตั้งใหม่ในเมือง ดูรูปที่ 6.6 นี้ คอมโพเนนต์จะเข้าถึงเมืองจากscreenPropsและจะใช้addLocation . ด้วย วิธีการจากscreenPropsเพื่อเพิ่มตำแหน่งให้กับเมือง

นำเข้าปฏิกิริยาจาก 'ปฏิกิริยา'
นำเข้า {
ดู,
ข้อความ,
สไตล์ชีต,
เลื่อนดู,

สัมผัสได้โดยไม่มีข้อเสนอแนะ
อินพุตข้อความ,
TouchableOpacity
} จาก 'react-native'
นำเข้า CenterMessage จาก '../components/CenterMessage'
นำเข้า { สี } จาก '../theme'
class City ขยาย React.Component {
navigationOptions แบบคงที่ = (อุปกรณ์ประกอบฉาก) => {
const { เมือง } = props.navigation.state.params
กลับ {
ชื่อเรื่อง: city.city,
headerTitleStyle: {
สี: 'ขาว',
ขนาดตัวอักษร: 20,
fontWeight: '400'
}
}
}
รัฐ = {
ชื่อ: '',
ข้อมูล:''
}
onChangeText = (คีย์ ค่า) => {
this.setState({
[คีย์]: ค่า
})
}
addLocation = () => {
if (this.state.name === '' || this.state.info === '') return
const { เมือง } = this.props.navigation.state.params
ตำแหน่ง const = {
ชื่อ: this.state.name,
ข้อมูล: this.state.info
}
this.props.screenProps.addLocation (ที่ตั้ง เมือง)
this.setState({ ชื่อ: '', ข้อมูล: '' })
}
เรนเดอร์ () {
const { เมือง } = this.props.navigation.state.params
กลับ (
<ดูสไตล์={{ ยืดหยุ่น: 1 }}>
สร้างการนำทางแบบคงที่Options
คุณสมบัติเช่นเดียวกับใน Cities.js
ทำลายวัตถุในเมือง

สร้างวัตถุตำแหน่งและ
เรียก this.props.screenProps
addLocation เพื่อเพิ่มสถานที่
และรีเซ็ตสถานะ
ทำลายเมือง
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 86


```
<ScrollView
contentContainerStyle={
[!city.locations.length && { ยืดหยุ่น: 1 }]
}>
<ดูสไตล์={[
style.locationsContainer,
!city.locations.length && { ยืดหยุ่น: 1,
justifyContent: 'ศูนย์' }
]}>
{
!city.locations.length &&
<CenterMessage message='ไม่มีสถานที่สำหรับเมืองนี้!' />
}
{
city.locations.map((ตำแหน่ง, ดัชนี) => (
<ดูคีย์={index} style={styles.locationContainer}>
<Text style={styles.locationName}>{location.name}</Text>
<Text style={styles.locationInfo}>{location.info}</Text>
</ดู>
))
}
</ดู>
</ScrollView>
<ป้อนข้อความ
onChangeText={val => this.onChangeText('ชื่อ', val)}
placeholder='ชื่อสถานที่'
ค่า={this.state.name}
style={styles.input}
ตัวยึดตำแหน่งTextColor='สีขาว'
/>
<ป้อนข้อความ
```

```
onChangeText={val => this.onChangeText('info', val)}
placeholder='ข้อมูลตำแหน่ง'
ค่า={this.state.info}
style={[styles.input, styles.input2]}
ตัวยึดตำแหน่งTextColor='สีขาว'
/>
<ดู style={styles.buttonContainer}>
<TouchableOpacity onPress={this.addLocation}>
<ดู style={styles.button}>
<Text style={styles.buttonText}>เพิ่มตำแหน่ง</Text>
</ดู>
</TouchableOpacity>
</ดู>
</ดู>
)
}
}
```

รหัสนี้สร้างคุณสมบัติnavigationOptionsก่อน คุณใช้ฟังก์ชันโทรกลับ
เพื่อส่งคืนวัตถุแทนที่จะประกาศวัตถุเพราะคุณต้องเข้าถึง
อุปกรณ์ประกอบฉากเพื่อเข้าถึงข้อมูลเมืองที่ส่งผ่านระบบนำทาง
คุณจำเป็นต้องทราบชื่อเมืองเพื่อใช้เป็นชื่อเส้นทางแทนสตริงที่ฮาร์ดโค้ด
addLocationวิธี destructures เมืองวัตถุที่มีอยู่จากthis.props
navigation.state.params เพื่อใช้ในภายหลังในฟังก์ชัน จากนั้นคุณสร้างสถานที่
แผนที่ทั่วเมืองใน
เมืองอาร์เรย์ และส่งกลับ a
องค์ประกอบที่แสดง
ชื่อเมืองและข้อมูล
สร้างแบบฟอร์ม
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 87

159
ข้อมูลที่คงอยู่
วัตถุที่มีชื่อตำแหน่งและข้อมูล กำลังเรียกthis.props.screenProps.addLoca-
tion เพิ่มตำแหน่งให้กับเมืองที่คุณกำลังดูอยู่ แล้วรีเซ็ตสถานะ อีกครั้ง,
ทำลายเมืองจากสถานะการนำทาง คุณต้องการเมืองเพื่อทำแผนที่เหนือตำแหน่ง-
ในเมืองและยังใช้เป็นข้อโต้แย้งในการสร้างที่ตั้งใหม่เพื่อระบุ

เมืองที่คุณกำลังอ้างอิง สุดท้าย คุณทำแผนที่เหนือเมือง ส่งคืนองค์ประกอบที่
แสดงทั้งชื่อเมืองและข้อมูลเมือง และสร้างแบบฟอร์มด้วยการป้อนข้อมูลสองข้อความ
และปุ่ม

6.3 ข้อมูลที่คงอยู่

คุณทำเสร็จแล้วและควรจะสามารถเรียกใช้แอปได้ เล่นกับแอพ เพิ่มเมือง
และสถานที่ แล้วรีเฟรช สังเกตว่าเมืองทั้งหมดจะหายไปเมื่อคุณรีเฟรช
เนื่องจากคุณจัดเก็บข้อมูลในหน่วยความจำเท่านั้น ลองใช้AsyncStorageไป
คงสถานะไว้ ดังนั้นหากผู้ใช้ปิดหรือรีเฟรชแอป ข้อมูลของพวกเขาจะยังคงมีอยู่
ในการดำเนินการดังกล่าว คุณจะต้องทำงานในคอมโพเนนต์แอปใน App.js และทำสิ่งต่อไปนี้:

- จัดเก็บอาร์เรย์เมืองในAsyncStorage ทุกครั้งที่มีการเพิ่มเมืองใหม่
- จัดเก็บอาร์เรย์เมืองในAsyncStorage ทุกครั้งที่มีการเพิ่มตำแหน่งใหม่ในเมือง
- เมื่อผู้ใช้เปิดแอป ให้ตรวจสอบAsyncStorage เพื่อดูว่ามีเมืองใดบ้าง
เก็บไว้ที่นั่น หากเป็นเช่นนั้น ให้อัปเดตสถานะด้วยเมืองเหล่านั้น
- AsyncStorage ยอมรับเฉพาะสตริงสำหรับค่าที่เก็บไว้ ดังนั้นเมื่อเก็บค่าโทร
JSON.stringify กับค่าหากไม่ใช่สตริงอยู่แล้ว และ JSON.parse หากคุณ
ต้องการแยกวิเคราะห์ค่าที่เก็บไว้ก่อนใช้งาน

เปิด App.js และทำการเปลี่ยนแปลง:

1 นำเข้าAsyncStorage และสร้างตัวแปรคีย์

```
นำเข้า {
#งดนำเข้าครั้งก่อน
AsyncStorage
} จาก 'react-native';
คีย์ const = 'สถานะ'
ส่งออกแอปคลาสเริ่มต้นขยายคอมโพเนนต์ {
#ละเว้นคำจำกัดความของคลาส
```

2 สร้างฟังก์ชันcomponentDidMount ที่จะตรวจสอบAsyncStorage และ get
รายการใดๆ ที่จัดเก็บไว้ที่นั่นด้วยค่าคีย์ที่คุณตั้งไว้:

```
async componentDidMount () {
ลอง {
ให้เมือง = รอ AsyncStorage.getItem (คีย์)
เมือง = JSON.parse(เมือง)
this.setState({ เมือง })
} จับ (จ) {
console.log ('ข้อผิดพลาดจาก AsyncStorage: ', e)
}
}
```

3 ในเมธอดaddCity จัดเก็บอาร์เรย์เมืองในAsyncStorage หลัง new
สร้างอาร์เรย์เมืองแล้ว:

```
addCity = (เมือง) => {
เมือง const = this.state.cities
city.push(เมือง)
this.setState({ เมือง })
AsyncStorage.setItem (คีย์ JSON.stringify (เมือง))
.then(() => console.log('storage updated!'))
.catch(e => console.log('e: ', e))
}
```

4 อัปเดตเมธอดaddLocation เพื่อจัดเก็บอาร์เรย์เมืองหลังจากตั้งค่าsetState แล้ว
เรียกว่า.

```
addLocation = (ที่ตั้ง, เมือง) => {
#รหัสก่อนหน้าถูกละเว้น
this.setState({
เมือง
}, () => {
AsyncStorage.setItem (คีย์ JSON.stringify (เมือง))
.then(() => console.log('storage updated!'))
.catch(e => console.log('e: ', e))
})
}
```

ตอนนี้ เมื่อผู้ใช้เปิดแอปหลังจากปิด ข้อมูลของพวกเขาจะยังคงมีอยู่

6.4 การใช้ DrawerNavigator เพื่อสร้าง Drawer-based การนำทาง

เราได้ศึกษาวิธีการสร้างการนำทางแบบสแต็กและแบบแท็บแล้ว ลองดูที่
API สำหรับสร้างการนำทางแบบลิ้นชัก
Drawer Navigator มี API ที่คล้ายกับของ stack และ tab navigator-
ทอร์ คุณจะใช้ฟังก์ชันcreateDrawerNavigatorจาก React Navigation ไปจนถึง Cre-
กินการนำทางแบบลิ้นชัก ขั้นแรกให้กำหนดเส้นทางที่จะใช้:

```
นำเข้า Page1 จาก './routeToPage1'
นำเข้า Page2 จาก './routeToPoage2'
```

ถัดไป กำหนดหน้าจอที่คุณต้องการใช้ในเนวิเกเตอร์:

```
หน้าจอ const = {
```

หน้า 1: { หน้าจอ: หน้า 1 },
หน้า 2: { หน้าจอ: หน้า 2 }
}
ตอนนี้คุณสามารถกำหนดเนวิเกเตอร์โดยใช้การกำหนดค่าหน้าจอและใช้ในแอพ:
const DrawerNav = createDrawerNavigator (หน้าจอ)
// ที่ไหนสักแห่งในแอปของเรา
<DrawerNav />
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 89

161
สรุป

สรุป

▪ ก่อนสร้างแอปพลิเคชัน ใช้เวลาวางกลยุทธ์ว่าต้องการให้จัดการอย่างไร
การนำทางและการกำหนดเส้นทาง
▪ ไลบรารีการนำทางจำนวนมากพร้อมใช้งานสำหรับ React Native แต่สองส่วนที่สำคัญที่สุดคือ
ommended คือ React Navigation และ React Native Navigation ตอบสนองการนำทาง
เป็นไลบรารีการนำทางที่ใช้ JavaScript และ React Native Navigation เป็นเนทีฟ
การดำเนินการ
▪ เนวิเกเตอร์มีสามประเภทหลัก:
– การนำทางแบบแท็บมักจะมีแท็บที่ด้านบนหรือด้านล่างของหน้าจอ
เมื่อคุณกดแท็บ คุณจะเข้าสู่หน้าจอที่สัมพันธ์กับแท็บนั้น สำหรับ
ตัวอย่างcreateBottomTabNavigatorสร้างแท็บที่ด้านล่างของหน้าจอ
– การเปลี่ยนการนำทางแบบสแต็กจากหน้าจอหนึ่งไปอีกหน้าจอหนึ่ง แทนที่
หน้าจอปัจจุบัน คุณสามารถย้อนกลับหรือเดินหน้าต่อไปในสแต็ก
การนำทางแบบกองซ้อนมักจะใช้การเปลี่ยนภาพเคลื่อนไหวบางประเภท
คุณสร้างการนำทางแบบสแต็กโดยใช้ฟังก์ชันcreateStackNavigator
– การนำทางแบบลิ้นชักมักจะเป็นเมนูที่โผล่ออกมาจาก
ด้านซ้ายหรือด้านขวาของหน้าจอและแสดงรายการตัวเลือก เมื่อคุณกดแป้น
ตัวเลือก ลิ้นชักจะปิดและคุณจะถูกนำไปที่หน้าจอใหม่ คุณสร้างการวาด-
การนำทางแบบEr -based โดยใช้ฟังก์ชันcreateDrawerNavigator
▪ ขึ้นอยู่กับประเภทของการนำทางที่คุณใช้—แบบแท็บ, แบบสแต็ก,
แบบลิ้นชักหรือแบบผสมกัน การกำหนดเส้นทางก็จะแตกต่างกันด้วย ทั้งหมด
เส้นทางหรือหน้าจอที่จัดการโดยไลบรารี React Navigation มีการนำทาง

prop ที่คุณสามารถใช้เพื่อควบคุมสถานะการนำทาง

- ใช้AsyncStorage เพื่อคงสถานะไว้ ดังนั้นหากผู้ใช้ปิดหรือรีเฟรชแอป

ข้อมูลยังคงมีอยู่

แอนิเมชั่น

บทนี้ครอบคลุม

- การสร้างแอนิเมชั่นพื้นฐานโดยใช้

Animated.timing

- การใช้การแก้ไขด้วยค่าภาพเคลื่อนไหว
- การสร้างภาพเคลื่อนไหวและแบบคู่ขนาน
- แอนิเมชั่นส่ายไปมาโดยใช้

Animated.stagger

- การใช้ไดรเวอร์ดั้งเดิมของแอนิเมชั่น oad

ไปยังเธรด UI ดั้งเดิม

สิ่งที่ยอดเยี่ยมอย่างหนึ่งเกี่ยวกับ React Native คือความสามารถในการสร้างแอนิเมชั่นได้อย่างง่ายดาย
โดยใช้ API แบบเคลื่อนไหว นี่เป็นหนึ่งใน React Native ที่เสถียรและใช้งานง่ายกว่า
API และเป็นหนึ่งในไม่กี่แห่งในระบบนิเวศ React Native ที่ไม่เหมือนพื้นที่
เช่นการนำทางและการจัดการของรัฐมีข้อตกลงเกือบ 100% เกี่ยวกับวิธีการ a
ปัญหาควรได้รับการแก้ไข
แอนิเมชั่นมักจะใช้เพื่อปรับปรุง UI ของแอปพลิเคชันและนำมาซึ่ง
มีชีวิตชีวามากขึ้นกับการออกแบบที่มีอยู่ บางครั้งความแตกต่างระหว่างค่าเฉลี่ยกับ
ประสบการณ์ผู้ใช้ที่สูงกว่าค่าเฉลี่ยสามารถนำมาประกอบกับการใช้แอนิเมชั่นที่เหมาะสมที่
เวลาที่เหมาะสม ซึ่งทำให้แอปแตกต่างจากแอปอื่นที่คล้ายคลึงกัน

163

ขอแนะนำ Animated API

กรณีการใช้งานจริงที่เรากล่าวถึงในบทนี้มีดังต่อไปนี้:

- ขยายอินพุตของผู้ใช้ที่เคลื่อนไหวเมื่อโฟกัส
- หน้าจอต้อนรับแบบเคลื่อนไหวที่มีชีวิตชีวามากกว่าหน้าจอต้อนรับแบบคงที่พื้นฐาน
- ตัวบ่งชี้การโหลดภาพเคลื่อนไหวแบบกำหนดเอง

ในบทนี้ เราจะเจาะลึกถึงวิธีการสร้างแอนิเมชั่น เราจะครอบคลุมทุกอย่าง
คุณจำเป็นต้องรู้เพื่อใช้ประโยชน์จาก Animated API อย่างเต็มที่

7.1 แนะนำ API แบบเคลื่อนไหว

Animated API มาพร้อมกับ React Native ดังนั้นหากต้องการใช้งาน สิ่งที่คุณต้องทำคือนำเข้า
เช่นเดียวกับที่คุณทำ React Native API หรือส่วนประกอบอื่น ๆ เมื่อสร้างแอนิเมชั่น
คุณต้องทำสี่สิ่งต่อไปนี้เสมอ:

1 นำเข้าภาพเคลื่อนไหวจาก React Native
2 สร้างค่าที่เคลื่อนไหวได้โดยใช้ Animated API
3 แนบค่ากับส่วนประกอบเป็นสไตล์
4 ทำให้ค่าเคลื่อนไหวได้โดยใช้ฟังก์ชัน

ส่วนประกอบที่เคลื่อนไหวได้สี่ประเภทจะมาพร้อมกับ Animated API นอกกรอบ:

- ดู
- ScrollView
- ข้อความ
- รูปภาพ

ตัวอย่างในบทนี้ทำงานเหมือนกันทุกประการในองค์ประกอบเหล่านี้
ในส่วนที่ 7.5 เรายังครอบคลุมถึงวิธีสร้างองค์ประกอบภาพเคลื่อนไหวแบบกำหนดเองโดยใช้ any
องค์ประกอบหรือส่วนประกอบที่มีcreateAnimatedComponent
มาดูกันสั้นๆ ว่าแอนิเมชั่นพื้นฐานอาจมีหน้าตาเป็นอย่างไรเมื่อใช้แอนิเมชั่น ใน
ตัวอย่าง คุณจะทำให้ระยะขอบด้านบนของกล่องเคลื่อนไหว (ดูรูปที่ 7.1)

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า {
สไตล์ชีต,
ดู,
เคลื่อนไหว
```

ปุ่ม
} จาก 'react-native';
ส่งออกคลาสเริ่มต้น RNAnimations ขยายคอมโพเนนต์ {
marginTop = ใหม่ Animated.Value(20);
เคลื่อนไหว = () => {
Animated.timing(
this.marginTop,
{
มูลค่า: 200,
ระยะเวลา: 500,
นำเข้า Animated API
จาก React Native
สร้างคุณสมบัติคลาสที่เรียกว่า
marginTop และกำหนดให้กับ an
ค่าภาพเคลื่อนไหวผ่านใน
ค่าเริ่มต้น (20 ในกรณีนี้)
สร้างฟังก์ชั่นที่จะทำให้ค่าเคลื่อนไหว
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 92

}
).เริ่ม();
}
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<ปุ่ม
title='กล่องเคลื่อนไหว'
onPress={this.animate}
/>
<Animated.View
style={[styles.box, { marginTop: this.marginTop }]} />
</ดู>
);
}
}
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {

ดิ้น: 1,
ช่องว่างภายใน: 10,
paddingTop: 50,
},
กล่อง: {
ความกว้าง: 150,
ความสูง: 150,
พื้นหลังสี: 'สีแดง'
}
});
แนบวิธีการเคลื่อนไหวกับ an
onPress handler เพื่อให้คุณสามารถเรียกมันว่า
ใช้องค์ประกอบ Animated.View แทน
ขององค์ประกอบการดูปกติ
ก่อนแอนิเมชั่น
หลังแอนิเมชั่น
รูปที่ 7.1 การสร้างภาพเคลื่อนไหวที่ขอบด้านบนของกล่องสี่เหลี่ยมโดยใช้ Animated
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 93

165
การทำให้อินพุตแบบฟอร์มเคลื่อนไหวเพื่อขยายโฟกัส
ตัวอย่างนี้ใช้ฟังก์ชันจับเวลาเพื่อทำให้ค่าเคลื่อนไหว ระยะเวลาการทำงานจะใช้เวลา
สองอาร์กิวเมนต์: ค่าเริ่มต้นและออบเจ็กต์การกำหนดค่า วัตถุการกำหนดค่าคือ
ผ่านtoValueเพื่อตั้งค่าที่แอนิเมชั่นควรเคลื่อนไหวและระยะเวลาใน
มิลลิวินาทีเพื่อกำหนดความยาวของภาพเคลื่อนไหว
แทนที่จะดูองค์ประกอบคุณใช้Animated.View ภาพเคลื่อนไหวมีสี่
ส่วนประกอบที่สามารถเคลื่อนไหวออกจากกล่อง: ดู , ภาพ , ScrollViewและข้อความ
ในการจัดสไตล์ของAnimated.Viewคุณจะส่งต่อสไตล์ต่างๆ ที่ประกอบด้วยฐาน
สไตล์ (styles.box) และสไตล์ภาพเคลื่อนไหว (marginTop)
ตอนนี้คุณได้สร้างองค์ประกอบภาพเคลื่อนไหวพื้นฐานแล้ว คุณจะสร้างส่วนประกอบเพิ่มเติมอีกสองสามรายการ
ภาพเคลื่อนไหวโดยใช้กรณีการใช้งานจริงที่อาจมีประโยชน์

7.2 การป้อนข้อมูลแบบฟอร์มเพื่อขยายโฟกัส

ในตัวอย่างนี้ คุณจะสร้างอินพุตแบบฟอร์มพื้นฐานที่ขยายเมื่อผู้ใช้เน้น
และสัญญาเมื่อข้อมูลเข้าเบลอ นี่เป็นรูปแบบ UI ยอดนิยม
พร้อมกับอุปกรณ์ประกอบฉากที่คุณเคยใช้กับองค์ประกอบTextInputในนี้

book เช่นvalue , placeholderและonChangeTextคุณยังสามารถใช้onFocusและ onBlur เพื่อเรียกใช้ฟังก์ชันเมื่อมีการโฟกัสและเบลออินพุต นั่นเป็นวิธีที่คุณจะ บรรลุแอนิเมชั่นนี้ (แสดงในรูปที่ 7.2)

นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า {
สไตล์ชีต,
ดู,
เคลื่อนไหว
ปุ่ม,
อินพุตข้อความ,
ข้อความ,
} จาก 'react-native';
ส่งออกคลาสเริ่มต้น RNAnimations ขยายคอมโพเนนต์ {
AnimatedWidth = ใหม่ Animated.Value(200);
เคลื่อนไหว = (ค่า) => {
Animated.timing(
this.animatedWidth,
{
toValue: ค่า,
ระยะเวลา: 750,
}
).เริ่ม()
}
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<Animated.View style={{ width: this.animatedWidth }}>
<ป้อนข้อความ
style={[styles.input]}
onBlur={() => this.animate(200)}
onFocus={() => this.animate(325)}
สร้างค่าเริ่มต้นสำหรับ
แอนิเมชั่น เรียกมันว่า แอนิเมชั่นWidth
สร้างฟังก์ชันเคลื่อนไหวที่จะทำให้เคลื่อนไหว
ค่าภาพเคลื่อนไหวของanimationWidth
แนบค่า animationWidth กับสไตล์ของ
คอนเทนเนอร์ View ถือองค์ประกอบอินพุต
แนบวิธีการเคลื่อนไหวกับ onBlur
และตัวจัดการ onFocus ส่งผ่านในที่ต้องการ
ความกว้างเมื่อแต่ละเหตุการณ์ถูกไล่ออก

หน้า 94


```
ref={input => this.input = อินพุต}
/>
</Animated.View>
<ปุ่ม
title='ส่ง'
onPress={() => this.input.blur()}
/>
</ดู>
);
}
}
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ดิ้น: 1,
ช่องว่างภายใน: 10,
paddingTop: 50,
},
ป้อนข้อมูล: {
ความสูง: 50,
ระยะขอบแนวนอน: 15,
backgroundColor: '#ededed',
ขอบบน: 10,
paddingแนวนอน: 9,
},
});
```

ก่อนแอนิเมชั่น

หลังแอนิเมชั่น

รูปที่ 7.2 การเคลื่อนไหวส่วนประกอบ TextInput เมื่ออินพุตถูกโฟกัส

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 95

7.3 การสร้างแอนิเมชั่นการโหลดแบบกำหนดเองโดยใช้การแก้ไข

หลายครั้ง คุณต้องสร้างแอนิเมชั่นที่วนซ้ำไม่รู้จบ เช่น การโหลด indicators และตัวชี้วัดกิจกรรม วิธีง่ายๆ วิธีหนึ่งในการสร้างแอนิเมชั่นดังกล่าวคือการใช้ฟังก์ชัน Animated.loop ในส่วนนี้ คุณใช้ Animated.loop ร่วมกับ Easing โมดูลเพื่อสร้างตัวบ่งชี้การโหลด หมุนรูปภาพในวงไม่สิ้นสุด!

จนถึงตอนนี้ เราเพิ่งดูการเรียกแอนิเมชันโดยใช้Animated.timingเท่านั้น ในเรื่องนี้ ตัวอย่างเช่น คุณต้องการให้แอนิเมชั่นทำงานอย่างต่อเนื่องโดยไม่หยุด เพื่อทำสิ่งนี้, คุณจะใช้วิธีการแบบคงใหม่ที่เรียกว่าห่วง Animated.loopเรียกใช้ภาพเคลื่อนไหวที่กำหนดทีละน้อย: ทุกครั้งที่ถึงจุดสิ้นสุด มันจะรีเซ็ตเป็นจุดเริ่มต้นและเริ่มต้นใหม่อีกครั้ง คุณยังจะต้องจัดการกับสไตล์ที่ต่างไปจากเดิมเล็กน้อย ในรายการ 7.1 และ 7.2 คุณใช้ค่าภาพเคลื่อนไหวโดยตรงในพร็อพสไตล์ของคอมโพเนนต์ ในส่วนย่อย-ตัวอย่างบ่อยครั้ง คุณจะเก็บค่าแอนิเมชั่นเหล่านี้ไว้ในตัวแปรและสอดแทรกค่าค่าก่อนที่จะใช้ตัวแปร interpolated ใหม่ในพร็อพสไตล์ เพราะคุณคือการสร้างเอฟเฟกต์การหมุน คุณจะใช้สตริงแทนตัวเลข ตัวอย่างเช่น คุณจะอ้างอิงค่าเช่น360degสำหรับสไตล์

Animated มีวิธีคลาสที่เรียกว่าinterpolateที่คุณสามารถใช้เพื่อจัดการ ani-ค่าที่จับคู่แล้วเปลี่ยนเป็นค่าอื่นที่คุณสามารถใช้ได้ การสอดแทรกวิธีรับวัตถุการกำหนดค่าด้วยสองปุ่ม: inputRange (array) และoutputRange (เป็นอาร์เรย์ด้วย) inputRangeคือค่าภาพเคลื่อนไหวดั้งเดิมที่คุณใช้งานในชั้นเรียน และ outputRange ระบุค่าที่ควรเปลี่ยนเป็นค่าดั้งเดิม

สุดท้าย คุณจะต้องเปลี่ยนค่าการค่อยๆ เปลี่ยนของภาพเคลื่อนไหว การค่อยๆ ลดลงจะช่วยให้คุณควบคุมการเคลื่อนไหวของแอนิเมชั่น ในตัวอย่างนี้ คุณต้องการการเคลื่อนไหวที่ราบรื่นและสม่ำเสมอสำหรับเอฟเฟกต์การหมุน ดังนั้นคุณจะใช้ฟังก์ชันการค่อยๆ เปลี่ยนเชิงเส้น

React Native มีวิธีใช้งานฟังก์ชันการค่อยๆ เปลี่ยนทั่วไปในตัว เช่นเดียวกับที่คุณได้นำเข้า API และส่วนประกอบอื่นๆ คุณสามารถนำเข้าโมดูล Easing และใช้งานได้พร้อมกับแอนิเมชั่น สามารถกำหนดค่าการค่อยๆ เปลี่ยนในออบเจ็กต์การกำหนดค่าที่คุณค่าชุดเช่นtoValueและระยะเวลาในอาร์กิวเมนต์ที่สองของAnimated.timing ดู Let 's ตัวอย่างมีมูลค่าการเคลื่อนไหวที่เรียกว่าanimatedMargin การตั้งค่าani-matedMargin เป็น 0 และการทำให้ค่าเป็น 200 เคลื่อนไหวได้ตามปกติจะบรรลุการผ่อนปรนผลโดยการสร้างภาพเคลื่อนไหวค่าระหว่าง 0 ถึง 200 โดยตรงในฟังก์ชันการจับเวลา โดยใช้การแก้ไข คุณสามารถทำให้ค่าเคลื่อนไหวระหว่าง 0 ถึง 1 ในฟังก์ชันการจับเวลาแทนได้และต่อมา interpolate ค่าโดยใช้วิธี Animated interpolate class sav-นำค่าไปใส่ในตัวแปรอื่น แล้วอ้างอิงตัวแปรนั้นในรูปแบบ

พันธมิตรในวิธีการเรนเดอร์:

```
const marginTop = animatedMargin.interpolate ({
InputRange: [0, 1],
outputRange: [0, 200],
});
```

ตอนนี้ ใช้การแก้ไขเพื่อสร้างตัวบ่งชี้การโหลด คุณจะแสดงตัวบ่งชี้เมื่อ
แอปพลิเคชั่นโหลด; ในcomponentDidMountคุณจะเรียกsetTimeoutซึ่งจะยกเลิก
สถานะการโหลดหลังจาก 2,000 มิลลิวินาที (ดูรูปที่ 7.3) ไอคอนที่ใช้ในที่นี้อยู่ที่
https://github.com/dabit3/react-native-in-action/blob/chapter7/assets/35633-200.png ;
ใช้หรือภาพอื่นๆ ได้ตามต้องการ
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 96


```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า {
ผ่อนปรน,
สไตล์ชีต,
ดู,
เคลื่อนไหว
ปุ่ม,
ข้อความ,
} จาก 'react-native';
ส่งออกคลาสเริ่มต้น RNAnimations ขยายคอมโพเนนต์ {
รัฐ = {
กำลังโหลด: จริง
}
componentDidMount () {
this.animate();
```

รูปที่ 7.3 การสร้างตัวแสดงการโหลดแบบหมุนโดยใช้การประมาณค่าและลูปแบบเคลื่อนไหว

เริ่มต้นสถานะด้วย a
ค่าการโหลดบูลีนของ true
เรียกภาพเคลื่อนไหวโดยเรียกสิ่งนี้
เคลื่อนไหวและเรียกใช้ setTimeout
ฟังก์ชั่นเพื่อตั้งค่าการโหลดเป็นเท็จใน

หน้า 97

169

การสร้างแอนิเมชั่นการโหลดแบบกำหนดเองโดยใช้การแก้ไข

```
setTimeout(() => this.setState({ กำลังโหลด: false }), 2000)
}
animationRotation = ใหม่ Animated.Value(0);
เคลื่อนไหว = () => {
Animated.loop(
Animated.timing(
this.animatedRotation,
{
มูลค่า: 1,
ระยะเวลา: 1800,
การค่อยๆ เปลี่ยน: Easing.linear,
}
)
).เริ่ม()
}
เรนเดอร์ () {
การหมุน const = this.animatedRotation.interpolate ({
InputRange: [0, 1],
outputRange: ['0deg', '360deg'],
});
const { กำลังโหลด } = this.state;
กลับ (
<ดู style={styles.container}>
{
กำลังโหลด ? (
<Animated.Image
source={require('./pathtoyourimage.png')}
สไตล์={{ ความกว้าง: 40,
ความสูง: 40,
แปลง: [{ หมุน: หมุน }] }}
/>
) : (
<Text>ยินดีต้อนรับ</Text>
)
}
</ดู>
);
```

```
}
}
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ดิ้น: 1,
justifyContent: 'ศูนย์',
alignItems: 'ศูนย์',
ช่องว่างภายใน: 10,
paddingTop: 50,
},
ป้อนข้อมูล: {
ความสูง: 50,
ระยะขอบแนวนอน: 15,
backgroundColor: '#ededed',
ขอบบน: 10,
paddingแนวนอน: 9,
},
});
```

ตั้งค่า animationRotation เริ่มต้นเป็น 0

สร้างเมธอดคลาสเคลื่อนไหวที่ผ่าน

Animated.timing เข้าสู่การโทรไปยัง Animated.loop

ใช้ค่า animationRotation เพื่อสร้างใหม่

ค่าการหมุนโดยใช้วิธีการสอดแทรก

ผ่านใน

แอนิเมชั่น

จุดเริ่มต้นและจุดสิ้นสุด

ค่า (0 และ 1)

ส่งผ่านค่าสำหรับ inputRange เพื่อจับคู่กับ

ตรวจสอบว่าการโหลดเป็นจริงหรือไม่

และตอบสนองตามนั้น

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 98

เคลื่อนไหววิธีการเรียนผ่านAnimated.timingเข้าโทรออกไปยังAnimated.loop ใน
การกำหนดค่าที่คุณตั้งtoValue 1, ระยะเวลาถึง 1800 และผ่อนคลายไปEasing.linear ,

เพื่อสร้างการเคลื่อนไหวที่ราบรื่น
animatedRotationค่าสร้างมูลค่าใหม่ที่เรียกว่าการหมุนโดยใช้
วิธีการสอดแทรก inputRange ให้ค่าเริ่มต้นและสิ้นสุดของภาพเคลื่อนไหว
และoutputRangeให้ค่าinputRangeควรจับคู่กับ: ค่าเริ่มต้นของ
0 องศาและค่าสุดท้าย 360 องศาสร้างการหมุน 360 องศาเต็มรูปแบบ
ในคำสั่งreturnให้ตรวจสอบก่อนว่าการโหลดเป็นจริงหรือไม่ ถ้าใช่ แสดงว่า
ตัวบ่งชี้การโหลดแบบเคลื่อนไหว (อัปเดตเส้นทางนี้เป็นเส้นทางของรูปภาพในแอปพลิเคชันของคุณ);
หากเป็นเท็จให้แสดงข้อความต้อนรับ แนบตัวแปรการหมุนเข้ากับการแปลง
หมุนค่าในการจัดแต่งทรงผมของAnimated.Image

7.4 การสร้างแอนิเมชั่นคู่ขนานหลายอัน

บางครั้งคุณจำเป็นต้องสร้างแอนิเมชั่นหลายๆ
อย่างลวกๆ ไลบรารี่เคลื่อนไหวมีวิธีการเรียนที่เรียกว่าขนานที่คุณสามารถใช้ทำ
นี้. Parallelเริ่มต้นอาร์เรย์ของภาพเคลื่อนไหวพร้อมกัน
ตัวอย่างเช่น เพื่อสร้างหน้าจอต้อนรับที่มีข้อความสองข้อความและปุ่มทั้งหมดปรากฏขึ้น
หากต้องการย้ายไปยังหน้าจอพร้อมกัน คุณสามารถสร้างแอนิเมชั่นสามแบบแยกกันและ
รูปที่ 7.4 หน้าจอต้อนรับโดยใช้แอนิเมชั่นคู่ขนาน
(แสดงหลังจากอนิเมชั่นเสร็จแล้ว)
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

171
การสร้างแอนิเมชั่นคู่ขนานหลายแบบ
โทร . start()กับแต่ละรายการ แต่วิธีที่มีประสิทธิภาพมากกว่าคือการใช้แอนิเมชั่น
ฟังก์ชั่นคู่ขนานและส่งผ่านอาร์เรย์ของแอนิเมชั่นให้ทำงานพร้อมกัน
ในตัวอย่างนี้ คุณจะสร้างหน้าจอต้อนรับที่เคลื่อนไหวในสองข้อความและ a
ปุ่มเมื่อส่วนประกอบติดตั้ง (ดูรูปที่ 7.4) เนื่องจากคุณใช้Animated
Parallel ภาพเคลื่อนไหวทั้งสามจะเริ่มพร้อมกันทุกประการ คุณจะเพิ่มความล่าช้า
คุณสมบัติในการกำหนดค่าเพื่อควบคุมเวลาเริ่มต้นของสองภาพเคลื่อนไหว

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า {
ผ่อนปรน,
สไตล์ชีต,
ดู,
```

```
เคลื่อนไหว
ข้อความ,
ไฮไลท์ที่สัมผัสได้,
} จาก 'react-native';
ส่งออกคลาสเริ่มต้น RNAnimations ขยายคอมโพเนนต์ {
AnimatedTitle = ใหม่ Animated.Value(-200);
animationSubtitle = ใหม่ Animated.Value (600);
AnimatedButton = ใหม่ Animated.Value (800);
componentDidMount () {
this.animate();
}
เคลื่อนไหว = () => {
ภาพเคลื่อนไหว.ขนาน([
Animated.timing(
this.animatedTitle,
{
มูลค่า: 200,
ระยะเวลา: 800,
}
),
Animated.timing(
this.animatedคำบรรยาย,
{
มูลค่า: 0,
ระยะเวลา: 1400,
ล่าช้า: 800,
}
),
Animated.timing(
this.animatedButton,
{
มูลค่า: 0,
ระยะเวลา: 1,000,
ล่าช้า: 2200,
}
)
]).เริ่ม();
}
```

เมื่อคุณสร้างชั้นเรียนด้วย

สร้างค่าภาพเคลื่อนไหวใหม่สามค่า

เรียกวิธีการเคลื่อนไหว ()

บน componentDidMount

โทร Animated.parallel และ

ผ่านในสาม Animated.timing

แอนิเมชั่นเพื่อเรียกทั้งสาม
แอนิเมชั่นที่จะเริ่มต้นในครั้งเดียว
โทร Animated.parallel และ
ผ่านในสาม Animated.timing
แอนิเมชั่นเพื่อเรียกทั้งสาม
แอนิเมชั่นที่จะเริ่มต้นในครั้งเดียว
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 100


```
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<สไตล์ข้อความเคลื่อนไหว={[styles.title,
{ marginTop: this.animatedTitle}]}>
ยินดีต้อนรับ
</Animated.Text>
<สไตล์ข้อความเคลื่อนไหว={[styles.subTitle,
{ marginLeft: this.animatedSubtitle }]}>
ขอบคุณสำหรับการเยี่ยมชมแอพของเรา!
</Animated.Text>
<Animated.View style={{ marginTop: this.animatedButton }}>
<TouchableHighlight style={styles.button}>
<Text>เริ่มต้น</Text>
</TouchableHighlight>
</Animated.View>
</ดู>
);
}
}
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ดิ้น: 1,
},
ชื่อ: {
textAlign: 'ศูนย์',
ขนาดตัวอักษร: 20,
ขอบด้านล่าง: 12,
},
```

```
หัวข้อย่อย: {
ความกว้าง: '100%',
textAlign: 'ศูนย์',
ขนาดตัวอักษร: 18,
ความทึบ: .8,
},
ปุ่ม: {
ขอบบน: 25,
พื้นหลังสี: '#ddd',
ความสูง: 55,
justifyContent: 'ศูนย์',
alignItems: 'ศูนย์',
ระยะขอบแนวนอน: 10,
}
});
```

7.5 การสร้างลำดับภาพเคลื่อนไหว

ลำดับภาพเคลื่อนไหวคือชุดของภาพเคลื่อนไหวที่เกิดขึ้นทีละรายการ
แอนิเมชั่นที่รอให้แอนิเมชั่นก่อนหน้าเสร็จสิ้นก่อนที่จะเริ่ม คุณสามารถ
สร้างลำดับภาพเคลื่อนไหวที่มีลำดับ เช่นเดียวกับคู่ขนาน , ลำดับใช้เวลาอาร์เรย์
ของแอนิเมชั่น:

```
Animated.sequence([
แอนิเมชั่นหนึ่ง,
```

แนบค่าภาพเคลื่อนไหวกับ
แต่ละองค์ประกอบที่คุณกำลังสร้างภาพเคลื่อนไหว

```
แอนิเมชั่นสอง,
แอนิเมชั่นสาม
]).เริ่ม()
```

ในตัวอย่างนี้ คุณจะสร้างลำดับที่ลดตัวเลข 1, 2 และ 3 ลงใน
หน้าจอ ห่างกัน 500 มิลลิวินาที (รูปที่ 7.5)

```
นำเข้า React { ส่วนประกอบ } จาก 'react';
นำเข้า {
สไตล์ชีต,
```

ดู,
เคลื่อนไหว
} จาก 'react-native';
ส่งออกคลาสเริ่มต้น RNAnimations ขยายคอมโพเนนต์ {
componentDidMount () {
this.animate();
}
รูปที่ 7.5 การสร้างลำดับภาพเคลื่อนไหวของตัวเลข
นำเข้าภาพเคลื่อนไหวจาก React Native
เรียกฟังก์ชันเคลื่อนไหว
เมื่อส่วนประกอบเมานต์
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 102

AnimatedValue1 = ใหม่ Animated.Value (-30);
AnimatedValue2 = ใหม่ Animated.Value (-30);
AnimatedValue3 = ใหม่ Animated.Value (-30);
เคลื่อนไหว = () => {
const createAnimation = (ค่า) => {
กลับ Animated.timing(
ค่า, {
มูลค่า: 290,
ระยะเวลา: 500
})
}
Animated.sequence([
createAnimation(this.AnimatedValue1),
createAnimation(this.AnimatedValue2),
createAnimation(นี้.AnimatedValue3)
]).เริ่ม()
}
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
<สไตล์ข้อความเคลื่อนไหว={[สไตล์.ข้อความ,
{ marginTop: this.AnimatedValue1}]}>
1
</Animated.Text>
<สไตล์ข้อความเคลื่อนไหว={[สไตล์.ข้อความ,

```
{ marginTop: this.AnimatedValue2}]}>
2
</Animated.Text>
<สไตล์ข้อความเคลื่อนไหว={[สไตล์.ข้อความ,
{ marginTop: this.AnimatedValue3}]}>
3
</Animated.Text>
</ดู>
);
}
}
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ดิ้น: 1,
justifyContent: 'ศูนย์',
flexDirection: 'แถว',
},
ข้อความ: {
ระยะขอบแนวนอน: 20,
ขนาดตัวอักษร: 26
}
});
```

ตัวอย่างนี้ใช้ค่าภาพเคลื่อนไหวเริ่มต้นที่ -30 เนื่องจากเป็นค่าmarginTop
ues สำหรับองค์ประกอบข้อความ: ข้อความถูกดึงออกจากด้านบนของหน้าจอและซ่อนไว้ก่อนหน้า
แอนิเมชั่นเริ่มต้นขึ้น createAnimationฟังก์ชั่นยังได้รับค่าภาพเคลื่อนไหว
เป็นข้อโต้แย้งของมัน
สร้างค่าภาพเคลื่อนไหวสามค่า
ส่งผ่าน -30 สำหรับค่าเริ่มต้น
สร้าง createAnimation
ทำหน้าที่เป็นตัวช่วยในการทำ
แอนิเมชั่นการจับเวลาใหม่
เริ่มลำดับ เรียก
createAnimation ครั้งเดียวสำหรับ
แต่ละค่าภาพเคลื่อนไหว
ส่งค่าภาพเคลื่อนไหวไปที่
ส่วนประกอบเคลื่อนไหวข้อความสามรายการ
ส่งค่าภาพเคลื่อนไหวไปที่
ส่วนประกอบเคลื่อนไหวข้อความสามรายการ

หน้า 103

175

ใช้ Animated.stagger เพื่อส่ายเวลาเริ่มแอนิเมชั่น

7.6 การใช้ Animated.stagger เพื่อทำให้แอนิเมชั่นเริ่มเดินเซ

ประเภทสุดท้ายของการเคลื่อนไหวที่เราจะไปกว่าเป็นAnimated.stagger เหมือนคู่ขนานและ
ลำดับ,โซเซใช้อาร์เรย์ของแอนิเมชั่น อาร์เรย์ของแอนิเมชั่นเริ่มต้นในพาร์-
allel แต่เวลาเริ่มต้นจะเซเท่าๆ กันในภาพเคลื่อนไหวทั้งหมด ต่างจากขนาน
และลำดับอาร์กิวเมนต์แรกที่ทำให้เซคือเวลาเซ และอาร์กิวเมนต์ที่สอง
ment คืออาร์เรย์ของแอนิเมชั่น:

```
Animated.stagger(
100,
[
แอนิเมชั่น1,
แอนิเมชั่น2,
แอนิเมชั่น3
]
).เริ่ม()
```

ในตัวอย่างนี้ คุณจะสร้างภาพเคลื่อนไหวจำนวนมากที่ใช้แบบไดนามิก
เพื่อเดินโซเซชุดกล่องสีแดงลงบนหน้าจอ (รูปที่ 7.6)

รูปที่ 7.6 การใช้ Animated.stagger เพื่อสร้าง an
อาร์เรย์ของแอนิเมชั่นที่เซ

หน้า 104

176


```
นำเข้า React { ส่วนประกอบ } จาก 'react'
นำเข้า {
สไตล์ชีต,
ดู,
เคลื่อนไหว
} จาก 'react-native'
ส่งออกคลาสเริ่มต้น RNAnimations ขยายคอมโพเนนต์ {
ตัวสร้าง () {
```

```
ซุปเปอร์()
this.animatedValues = []
สำหรับ (ให้ i = 0; i < 1000; i++) {
this.animatedValues[i] = ใหม่ Animated.Value(0)
}
this.animations = this.animatedValues.map (ค่า => {
กลับ Animated.timing(
ค่า,
{
มูลค่า: 1,
ระยะเวลา: 6000
}
)
})
}
componentDidMount () {
this.animate()
}
เคลื่อนไหว = () => {
Animated.stagger(15, this.animations).start()
}
เรนเดอร์ () {
กลับ (
<ดู style={styles.container}>
{
this.animatedValues.map((ค่า, ดัชนี) => (
<คีย์ Animated.View={index}
style={[{ความทึบ: ค่า},
style.box]} />
))
}
</ดู>
);
}
}
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ดิ้น: 1,
justifyContent: 'ศูนย์',
flexDirection: 'แถว',
flexWrap: 'ห่อ'
},
กล่อง: {
```

นำเข้าภาพเคลื่อนไหวจาก React Native

สร้างอาร์เรย์

ที่จะบรรจุ 1,000
ค่าภาพเคลื่อนไหวของ0
สร้างอาร์เรย์ของ
Animated.timing
แอนิเมชั่น
อ้างถึง
ค่าภาพเคลื่อนไหว
สร้างขึ้นใน
อาร์เรย์ค่าภาพเคลื่อนไหว
เรียกวิธีการเคลื่อนไหว
เรียก Animated.stagger().start(), ส่งผ่าน
เวลา 15 ms และอาร์เรย์ของ
แอนิเมชั่น
แมปทับแอนิเมชั่น สร้าง an
Animated.View สำหรับแต่ละรายการในอาร์เรย์
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

177
เคล็ดลับที่เป็นประโยชน์อื่น ๆ สำหรับการใช้ไลบรารีแบบเคลื่อนไหว
ความกว้าง: 15,
ความสูง: 15,
ระยะขอบ: .5,
พื้นหลังสี: 'สีแดง'
}
})

7.7 เคล็ดลับที่เป็นประโยชน์อื่น ๆ สำหรับการใช้ห้องสมุดเคลื่อนไหว

นอกเหนือจากส่วนต่างๆ ของ Animated API ที่เราได้กล่าวถึงไปแล้ว ยังมีเทคโนโลยีอื่นๆ อีกสองสามอย่าง
niques มีประโยชน์ในการรู้เกี่ยวกับ: การรีเซ็ตค่าภาพเคลื่อนไหว, การเรียกใช้การเรียกกลับ, การปิด-
กำลังโหลดแอนิเมชั่นไปยังเธรดดั้งเดิม และสร้างส่วนประกอบแอนิเมชั่นแบบกำหนดเองได้
ส่วนนี้จะพิจารณาอย่างรวดเร็วในแต่ละส่วนเหล่านี้

7.7.1 การรีเซ็ตค่าภาพเคลื่อนไหว

หากคุณกำลังเรียกแอนิเมชั่น คุณสามารถรีเซ็ตค่าเป็นสิ่งที่คุณต้องการโดยใช้
setValue (ค่า) สิ่งนี้มีประโยชน์หากคุณได้เรียกแอนิเมชั่นด้วยค่าและ

ต้องเรียกแอนิเมชั่นอีกครั้ง และคุณต้องการรีเซ็ตค่าเป็นค่าเดิม
ค่าหรือค่าใหม่:

```
เคลื่อนไหว = () => {
this.animatedValue.setValue(300);
#มาต่อที่นี่กับค่าอนิเมชั่นใหม่
}
```

7.7.2 เรียกเรียกกลับ

เมื่อแอนิเมชั่นเสร็จสิ้น ฟังก์ชันเรียกกลับเสริมสามารถเริ่มทำงานได้เช่น
แสดงที่นี่:

```
Animated.timing(
this.animatedValue,
{
มูลค่า: 1,
ระยะเวลา: 1,000
}
).start(() => console.log('ภาพเคลื่อนไหวเสร็จสมบูรณ์!'))
```

7.7.3 การถ่ายแอนิเมชั่นไปยังเธรดดั้งเดิม

ไลบรารี Animated ดำเนินการสร้างภาพเคลื่อนไหวโดยใช้เธรด JavaScript นอกกรอบ
ในกรณีส่วนใหญ่ วิธีนี้ใช้ได้ดี และคุณไม่ควรมีปัญหาด้านประสิทธิภาพมากนัก
แต่ถ้ามีอะไรมาบล็อกเธรด JavaScript คุณอาจพบปัญหาเช่น เฟรมกำลัง
ข้าม ทำให้แอนิเมชั่นกระตุกหรือกระตุก
มีวิธีใช้เธรด JavaScript: คุณสามารถใช้การกำหนดค่า
บูลีนที่เรียกว่าuseNativeDriver useNativeDriver ถ่ายแอนิเมชั่นไปที่
เธรด UI ดั้งเดิม และโค้ดเนทีฟสามารถอัปเดตมุมมองได้โดยตรงบน UI
เธรดดังแสดงที่นี่:

```
Animated.timing(
this.animatedValue,
```

หน้า 106


```
{
มูลค่า: 100,
ระยะเวลา: 1,000,
useNativeDriver: จริง
}
```

).เริ่ม();
ไม่ใช่ทุกอนิเมชั่นที่สามารถถ่ายออกได้โดยใช้useNativeDriverดังนั้นอย่าลืมตรวจสอบ
เอกสาร API แบบเคลื่อนไหวเมื่อคุณใช้งาน ในขณะที่เขียนนี้ เฉพาะอุปกรณ์ที่ไม่ใช่โครงร่าง
erties สามารถเคลื่อนไหวได้โดยใช้วิธีนี้ คุณสมบัติ flexbox เช่นเดียวกับคุณสมบัติ like
ระยะขอบและช่องว่างภายในไม่สามารถเคลื่อนไหวได้

7.7.4 การสร้างองค์ประกอบที่เคลื่อนไหวได้เองโดยใช้ createAnimatedComponent

เราได้กล่าวถึงในหัวข้อที่ 7.1 ว่าส่วนประกอบที่เคลื่อนไหวได้เพียงอย่างเดียวที่ออกมาจากกล่องคือ
ดู,ข้อความ,ภาพ, และScrollView นอกจากนี้ยังมีวิธีสร้างส่วนประกอบที่เป็นภาพเคลื่อนไหวอีกด้วย
จากองค์ประกอบหรือองค์ประกอบ React Native ที่มีอยู่หรือแบบกำหนดเอง คุณสามารถทำได้โดย
ห่อองค์ประกอบในการเรียกไปยังcreateAnimatedComponent นี่คือตัวอย่าง:

```
ปุ่ม const = Animated.createAnimatedComponent (TouchableHighlight)
<ปุ่ม onPress={somemethod} style={styles.button}>
<Text>สวัสดีชาวโลก</Text>
</ปุ่ม>
```

ตอนนี้คุณสามารถใช้ปุ่มได้เหมือนกับส่วนประกอบ React Native ทั่วไป

สรุป

- API แบบเคลื่อนไหวในตัวเป็นวิธีที่แนะนำในการสร้างแอนิเมชั่นใน
ตอบโต้พื้นเมือง
- Animated.timing เป็นวิธีการหลักในการสร้างแอนิเมชั่นโดยใช้ปุ่ม
ห้องสมุดเคลื่อนไหว
- ส่วนประกอบเดียวที่เคลื่อนไหวได้นอกกรอบคือView , Text ,
ScrollView และ Image แต่คุณสามารถสร้างส่วนประกอบที่เคลื่อนไหวได้เอง
ใช้createAnimatedComponent
- ในการสอดแทรกและนำค่าภาพเคลื่อนไหวมาใช้ใหม่ ให้ใช้ Animated interpolate
กระบวนการ.
- ในการสร้างและทริกเกอร์อาร์เรย์ของแอนิเมชั่นในเวลาเดียวกัน ให้ใช้Animated
.ขนาน
- หากต้องการสร้างแอนิเมชั่นวนซ้ำไม่รู้จบให้ใช้Animated.loop
- ใช้Animated.sequence เพื่อสร้างลำดับของแอนิเมชั่นที่ดำเนินการอย่างใดอย่างหนึ่ง
หลังจากนั้นอีก
- ใช้Animated.stagger เพื่อสร้างอาร์เรย์ของแอนิเมชั่นที่เกิดขึ้นพร้อมกัน
แต่เวลาเริ่มต้นจะถูกเซตามเวลาที่ผ่านไป

179

การใช้ข้อมูล Redux ห้องสมุดสถาปัตยกรรม

บทนี้ครอบคลุม

- API บริบทของ React ทำงานอย่างไร
- การสร้างร้านRedux
- วิธีใช้การกระทำของ Redux และตัวลด to

จัดการสถานะโลก

- ลดองค์ประกอบโดยใช้combineReducers

เมื่อสร้างแอปพลิเคชัน React และ React Native ในโลกจริง คุณจะทำได้อย่างรวดเร็ว

เรียนรู้ว่าชั้นข้อมูลอาจซับซ้อนและจัดการไม่ได้หากไม่ได้รับการจัดการ

อย่างแม่นยำและตั้งใจมาก วิธีหนึ่งในการจัดการข้อมูลคือเก็บไว้ในส่วนประกอบ
ระบุและส่งต่อเป็นอุปกรณ์ประกอบฉากดังที่เราทำตลอดทั้งเล่มนี้ อีกทางหนึ่ง
คือการใช้รูปแบบสถาปัตยกรรมข้อมูลหรือไลบรารี่ บทนี้ครอบคลุมไลบรารี Redux:
เป็นวิธีที่ใช้กันอย่างแพร่หลายในการจัดการข้อมูลในระบบนิเวศ React และมันคือ
ดูแลโดย Facebook ซึ่งเป็นทีมเดียวกับที่ดูแลทั้ง React และ React Native

8.1 Redux คืออะไร?

ในเอกสารประกอบของ Redux ไลบรารีถูกอธิบายว่าเป็น “สถานะที่คาดการณ์ได้
tainer สำหรับแอป JavaScript” Redux นั้นเป็นอ็อบเจกต์สถานะโกลบอลที่เป็นซิงเกิล
แหล่งที่มาของความจริงในใบสมัคร ได้รับวัตถุสถานะส่วนกลางนี้เป็นอุปกรณ์ประกอบฉากใน
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

ตอบสนององค์ประกอบดั้งเดิม ทุกครั้งที่มีการเปลี่ยนแปลงข้อมูลในสถานะ Redux ค่า
แอปพลิเคชันทั้งหมดได้รับข้อมูลใหม่นี้เป็นอุปกรณ์ประกอบฉาก
Redux ช่วยลดความซับซ้อนของสถานะแอปพลิเคชันโดยการย้ายทั้งหมดมาไว้ในที่เดียวที่เรียกว่าร้านค้า นี้
ทำให้ง่ายต่อการให้เหตุผลและเข้าใจ เมื่อคุณต้องการค่าของ
บางอย่าง คุณจะรู้ว่าต้องดูที่ไหนในแอปพลิเคชัน Redux และสามารถคาดหวังได้
ค่าเดียวกันที่จะพร้อมใช้งานและเป็นปัจจุบันในที่อื่นในแอปพลิเคชันด้วย
Redux ทำงานอย่างไร? ใช้ประโยชน์จากคุณลักษณะ React ที่เรียกว่าบริบท a
กลไกการสร้างและจัดการสถานะโลก

8.2 การใช้บริบทเพื่อสร้างและจัดการสถานะโลก ในแอปพลิเคชัน React

Context คือ React API ที่สร้างตัวแปรส่วนกลางที่สามารถเข้าถึงได้ทุกที่ใน
แอปพลิเคชัน ตราบใดที่ส่วนประกอบที่ได้รับบริบทเป็นลูกของคอม-
ponent ที่สร้างมัน โดยปกติคุณจะต้องทำสิ่งนี้โดยส่งอุปกรณ์ประกอบฉากลงในแต่ละระดับ
ของโครงสร้างส่วนประกอบ ด้วยบริบท คุณไม่จำเป็นต้องใช้อุปกรณ์ประกอบฉาก คุณสามารถใช้ได้
บริบทที่ใดก็ได้ในแอปและเข้าถึงได้โดยไม่ต้องส่งต่อไปยังแต่ละระดับ
หมายเหตุแม้ว่าบริบทจะดีที่จะเข้าใจและถูกนำมาใช้ในการเปิดจำนวนมาก
ไลบรารีต้นทาง คุณอาจไม่จำเป็นต้องใช้มันในแอพ เว้นแต่ว่าคุณกำลังสร้าง
ไลบรารีโอเพ่นซอร์สหรือไม่สามารถหาวิธีอื่นในการแก้ไขปัญหาได้ เรากำลังค-

ด่ามันเพื่อให้คุณเข้าใจอย่างถ่องแท้ว่า Redux ทำงานอย่างไรภายใต้ประทุน
มาดูวิธีการสร้างบริบทในโครงสร้างพื้นฐานขององค์ประกอบสามองค์ประกอบ
nents: ผู้ปกครอง , Child1และchild2 ตัวอย่างนี้แสดงวิธีการใช้ทั้งแอปพลิเคชัน
ธีมจากระดับผู้ปกครอง ซึ่งทำให้สามารถควบคุมสไตล์ของ an
แอปพลิเคชันทั้งหมดหากจำเป็น

```
const ThemeContext = React.createContext ()
คลาสผู้ปกครองขยายส่วนประกอบ {
รัฐ = { themeValue: 'light' }
toggleThemeValue = () => {
ค่า const = this.state.themeValue === 'มืด' ? 'สว่างมืด'
this.setState ({ themeValue: ค่า })
}
เรนเดอร์ () {
กลับ (
<ThemeContext.Provider
ค่า={{
ค่าธีม: this.state.themeValue,
toggleThemeValue: this.toggleThemeValue
}}
>
<ดู style={styles.container}>
```

สร้างตัวแปรใหม่ชื่อ ThemeContext
สร้างสถานะ themeValue
ตัวแปรที่มีค่า 'แสง'
ตรวจสอบค่าธีมปัจจุบัน
และสลับเป็น 'สว่าง' หรือ 'มืด'
ให้
บริบทสำหรับเด็ก
ส่วนประกอบ
อะไรก็ตาม
ห่อด้วย
ผู้ให้บริการคือ
พร้อมที่จะ
ลูก ๆ ของ
ส่วนประกอบใน
ผู้บริโภค

หน้า 109

181

การนำ Redux ไปใช้ด้วย React Native app

```
<Text>สวัสดีชาวโลก</Text>
</ดู>
<Child1/>
</ThemeContext.Provider>
);
}
}
const Child1 = () => <Child2 />
const Child2 = () => (
<ThemeContext.Consumer>
{(วาล) => (
<ดูสไตล์={[styles.container,
val.themeValue === 'มืด' &&
{ backgroundColor: 'black' }]}>
<Text style={styles.text}>สวัสดีจาก Component2</Text>
<รูปแบบข้อความ={styles.text}
onPress={val.toggleThemeValue}>
สลับค่าธีม
</Text>
</ดู>
)}
</ThemeContext.Consumer>
)
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ดิ้น: 1,
justifyContent: 'ศูนย์',
alignItems: 'ศูนย์',
พื้นหลังสี: '#F5FCFF',
},
ข้อความ: {
ขนาดตัวอักษร: 22,
สี: '#666'
}
})
```

child2ไร้สัญชาติฟังก์ชันส่งกลับองค์ประกอบที่ห่อเป็นThemeContext.Consumer ThemeContext.Consumer ต้องการฟังก์ชันที่เป็นลูกของมัน ฟังก์ชัน ได้รับอาร์กิวเมนต์ที่มีบริบทใดก็ตามที่มี (ในกรณีนี้

val วัตถุที่มีคุณสมบัติสองอย่าง) ตอนนี้คุณสามารถใช้ค่าบริบทใน
ส่วนประกอบ.
เมื่อคุณใช้ Redux กับ React คุณจะใช้ประโยชน์จากฟังก์ชันที่เรียกว่าconnect ,
ซึ่งโดยพื้นฐานแล้วใช้บริบทบางส่วนและทำให้พร้อมใช้งานเป็นอุปกรณ์ประกอบฉาก
เน็ท การทำความเข้าใจบริบทควรทำให้การเรียนรู้ Redux ง่ายขึ้นมาก!

8.3 การใช้ Redux ด้วยแอป React Native

ตอนนี้คุณก็รู้พื้นฐานของว่า Redux คืออะไรและเห็นว่าเกิดอะไรขึ้น
ภายใต้ประทุนพร้อมบริบท มาสร้างแอป React Native ใหม่และเริ่มเพิ่ม
ฟังก์ชันไร้สัญชาติที่ส่งคืนส่วนประกอบ
แสดงว่าคุณไม่ได้ผ่านอุปกรณ์ประกอบฉาก
ระหว่างผู้ปกครองและเด็ก2
ฟังก์ชันไร้สัญชาติที่ส่งคืนส่วนประกอบ
ห่อด้วย ThemeContext.Consumer
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

รีดักซ์ คุณจะต้องสร้างแอปรายการพื้นฐานที่คุณสามารถใช้ติดตามหนังสือที่คุณอ่านได้
(ดูรูปที่ 8.1) ทำตามขั้นตอนเหล่านี้:

1 สร้างแอปพลิเคชัน React Native ใหม่และเรียกมันว่า RNRedux:

```
react-native init RNRedux
```

2 เปลี่ยนเป็นไดเร็กทอรีใหม่:

```
cd RNRedux
```

3 ติดตั้งการพึ่งพาเฉพาะ Redux ที่คุณต้องการ:

```
npm ฉัน redux react-redux –-save
```

4 ในรูทของไดเร็กทอรี ให้สร้างโฟลเดอร์ชื่อ src และเพิ่มเข้าไปดังต่อไปนี้
ไฟล์: Books.js และ actions.js นอกจากนี้ใน src ให้สร้างโฟลเดอร์ชื่อ reducers ประกอบด้วย-
ing สองไฟล์: bookReducer.js และ index.js โครงสร้างโฟลเดอร์ src ควรตอนนี้
มีลักษณะเหมือนรูปที่ 8.2
สิ่งต่อไปที่ต้องทำคือสร้างสถานะ Redux ชิ้นแรก คุณจะทำสิ่งนี้ใน
bookReducer.js ในหัวข้อ 8.1 ฉันอธิบายว่า Redux เป็นวัตถุส่วนกลาง เพื่อสร้างสิ่งนี้
วัตถุส่วนกลาง คุณจะประกอบวัตถุขนาดเล็กเข้าด้วยกัน โดยใช้สิ่งที่เรียกว่าตัวลดขนาด

รูปที่8.1 กรอกใบสมัครรายการหนังสือ
รูปที่ 8.2 โครงสร้างโฟลเดอร์ RNRedux src
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

183
การสร้าง Redux reducers เพื่อเก็บสถานะ Redux

8.4 การสร้างตัวลด Redux เพื่อคงสถานะ Redux

รีดิวเซอร์เป็นฟังก์ชันที่ส่งคืนอ็อบเจ็กต์ เมื่อรวมกับรีดิวซ์อื่นๆ พวกมัน
สร้างสถานะโลก ตัวลดสามารถคิดได้ง่ายขึ้นว่าเป็นที่เก็บข้อมูล แต่ละร้าน
มีข้อมูลบางส่วนซึ่งเป็นสิ่งที่ตัวลดขนาดทำในสถาปัตยกรรม Redux
ในโฟลเดอร์ reducers มีไฟล์สองไฟล์: bookReducer.js และ index.js ใน index.js คุณจะ
รวมตัวลดทั้งหมดในแอปเพื่อสร้างสถานะทั่วโลก แอพจะมีเพียง
ตัวลดหนึ่งตัวเริ่มต้นด้วย (bookReducer) ดังนั้นวัตถุสถานะส่วนกลางจะมีลักษณะบางอย่าง
แบบนี้:

```
{
ตัวลดหนังสือ: {}
}
```

คุณได้ยังไม่ได้ตัดสินใจว่าจะใส่อะไรในbookReducer อาร์เรย์สำหรับจัดเก็บรายการหนังสือ
จะเป็นการเริ่มต้นที่ดี ตัวลดนี้จะสร้างและส่งคืนสถานะที่คุณจะเข้าถึงได้
ต่อมาจากร้าน Redux ใน reducers/bookReducer.js ให้สร้างตัวลดตัวแรกของคุณ นี้
รหัสสร้างฟังก์ชันที่มีจุดประสงค์เพียงอย่างเดียว (สำหรับตอนนี้) คือการคืนสถานะ

```
const initialState = { #A
หนังสือ: [{ ชื่อ: 'East of Eden' ผู้แต่ง: 'John Steinbeck' }]
} #NS
const bookReducer = (สถานะ = initialState) => {
คืนสถานะ
}
ส่งออก bookReducer เริ่มต้น
```

initialStateวัตถุจะถือเป็นรัฐที่จุดเริ่มต้น ในกรณีนี้ นั่นคืออาร์เรย์ของ
หนังสือที่คุณจะเติมด้วยวัตถุที่มีชื่อและอุปกรณ์ประกอบฉากของผู้เขียน คุณสร้าง
ฟังก์ชันที่รับอาร์กิวเมนต์stateและตั้งค่าเริ่มต้นเป็นสถานะเริ่มต้น เมื่อไหร่
ฟังก์ชันนี้ถูกเรียกก่อนสถานะจะไม่ถูกกำหนด และจะคืนค่าinitialState
วัตถุ. ในขณะนี้ จุดประสงค์เดียวของฟังก์ชันคือการคืนสถานะ

ตอนนี้คุณได้สร้างตัวลดขนาดตัวแรกแล้ว ให้ไปที่rootReducer (reducers/index.js)
และสร้างสิ่งที่จะเป็นสถานะโลก ตัวลดรากรวบรวมตัวลดทั้งหมดใน
แอปพลิเคชันและช่วยให้คุณสร้างร้านค้าระดับโลก (วัตถุสถานะ) โดยการรวมเข้าด้วยกัน

นำเข้า { combineReducers } จาก 'redux'
นำเข้า bookReducer จาก './bookReducer'
const rootReducer = รวมตัวลด ({
bookReducer
})
ส่งออกเริ่มต้น rootReducer
ต่อไป ในการรวมสิ่งนี้เข้าด้วยกัน คุณจะต้องไปที่ App.js สร้างร้าน Redux และสร้าง
เก็บพร้อมใช้งานสำหรับส่วนประกอบย่อยทั้งหมดโดยใช้ตัวช่วย Redux และ React-Redux
สร้างอ็อบเจ็กต์ initialState
รับอาร์กิวเมนต์ของรัฐและตั้งค่า
ค่าเริ่มต้นเป็นสถานะเริ่มต้น
คืนสถานะ
นำเข้า CombineReducers
ฟังก์ชันจาก Redux
นำเข้า
bookReducer
ลด
สร้างตัวลดรากที่มี
ตัวลดทั้งหมด ในกรณีนี้ประกอบด้วย
the single property bookReducer

หน้า 112

8.5 การเพิ่มผู้ให้บริการและสร้างร้านค้า

ในส่วนนี้ คุณจะต้องเพิ่มผู้ให้บริการลงในแอป ผู้ให้บริการมักจะเป็นองค์ประกอบหลัก
ที่ส่งข้อมูลบางอย่างไปยังส่วนประกอบย่อยทั้งหมด ใน Redux ผู้ให้บริการจะผ่าน
สถานะทั่วโลก/ร้านค้าไปยังส่วนที่เหลือของแอปพลิเคชัน ใน App.js ให้อัปเดตโค้ดดังนี้

นำเข้าปฏิกิริยาจาก 'ปฏิกิริยา'
นำเข้าหนังสือจาก './src/Books'
นำเข้า rootReducer จาก './src/reducers'
นำเข้า { ผู้ให้บริการ } จาก 'react-redux'
นำเข้า { createStore } จาก 'redux'
เก็บ const = createStore (rootReducer)
ส่งออกแอปคลาสเริ่มต้นขยาย React.Component {
เรนเดอร์ () {
กลับ (
<ร้านผู้ให้บริการ={ร้านค้า} >
<หนังสือ />
</ผู้ให้บริการ>
)
}
}
รูปที่ 8.3 แสดงรายชื่อหนังสือจาก
ร้าน Redux
นำเข้าส่วนประกอบหนังสือ
(สร้างในรายการ 8.5)
นำเข้า rootReducer
นำเข้า wrapper ของผู้ให้บริการจาก react-redux
นำเข้า createStore
สร้างร้านผ่าน
ในรูทReducer
ส่งกลับองค์ประกอบหนังสือที่ห่อ
ในองค์ประกอบผู้ให้บริการส่งผ่านใน
เก็บไว้เป็นพร็อพให้กับผู้ให้บริการ
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 113

185
การเข้าถึงข้อมูลโดยใช้ฟังก์ชันเชื่อมต่อ
ผู้ให้บริการเสื้อคลุมจะใช้ในการห่อองค์ประกอบหลัก ลูกของผู้ให้บริการ
จะมีสิทธิ์เข้าถึงร้าน Redux createStoreเป็นยูทิลิตี้จาก Redux ที่คุณใช้ถึง
สร้างร้านค้า Redux โดยผ่านในrootReducer คุณเสร็จสิ้นด้วยพื้นฐาน
การตั้งค่า Redux และตอนนี้คุณสามารถเข้าถึงร้าน Redux ในแอปได้แล้ว

ในหนังสือองค์ประกอบที่คุณจะขอเข้าสู่ร้านค้า Redux, ดึงออกหนังสืออาร์เรย์
และแผนที่เหนือหนังสือ แสดงใน UI (รูปที่ 8.3) เพราะหนังสือเป็นเด็ก
ของProviderสามารถเข้าถึงอะไรก็ได้ในร้าน Redux

8.6 การเข้าถึงข้อมูลโดยใช้ฟังก์ชันเชื่อมต่อ

คุณเข้าถึงร้าน Redux จากองค์ประกอบลูกโดยใช้ฟังก์ชันการเชื่อมต่อ
จาก react-redux อาร์กิวเมนต์แรกที่เชื่อมต่อคือฟังก์ชันที่ให้คุณเข้าถึง
สถานะ Redux ทั้งหมด จากนั้นคุณสามารถส่งคืนวัตถุด้วยชิ้นส่วนของร้านค้าได้
คุณต้องการเข้าถึง
connect เป็นฟังก์ชัน curried ความหมายในความหมายพื้นฐานที่สุดคือฟังก์ชันที่ส่งคืน
ฟังก์ชั่นอื่น คุณจะมีอาร์กิวเมนต์สองชุด และพิมพ์เขียวที่ดูบางอย่าง-
สิ่งเช่นนี้การเชื่อมต่อ (args) (args) คุณสมบัติในวัตถุที่ส่งคืนจาก
อาร์กิวเมนต์แรกที่เชื่อมต่อจะพร้อมใช้งานสำหรับส่วนประกอบเป็นอุปกรณ์ประกอบฉาก
มาดูกันว่ามันหมายความว่าอย่างไรโดยดูที่ฟังก์ชันการเชื่อมต่อที่คุณจะใช้ในหนังสือ
ส่วนประกอบ js

```
เชื่อมต่อ(
(รัฐ) => {
กลับ {
หนังสือ: state.bookReducer.books
}
}
)(หนังสือ)
```

อาร์กิวเมนต์แรกที่เชื่อมต่อคือฟังก์ชันที่ให้วัตถุสถานะ Redux ทั่วโลกเป็น
อาร์กิวเมนต์ จากนั้นคุณสามารถอ้างอิงอ็อบเจ็กต์สถานะนี้และเข้าถึงอะไรก็ได้ใน
รัฐ Redux คุณส่งคืนวัตถุจากฟังก์ชันนี้ กุญแจไหนถูกคืน
ในวัตถุจะพร้อมใช้งานเป็นอุปกรณ์ประกอบฉากในส่วนประกอบที่คุณกำลังห่อ: ในนี้
กรณี, หนังสือ . คุณส่งผ่านหนังสือเป็นอาร์กิวเมนต์เดียวไปยังฟังก์ชันการเชื่อมต่อที่สอง
เรียกใช้ฟังก์ชัน
บ่อยครั้ง คุณจะแยกฟังก์ชันนี้และเก็บไว้ในตัวแปรเพื่อให้อ่านง่ายขึ้น:

```
const mapStateToProps = สถานะ => ({
หนังสือ: state.bookReducer.books
})
```

ในองค์ประกอบที่เชื่อมต่อนี้เป็นคุณสมบัติใหม่ที่เรียกว่าthis.props.booksซึ่งก็คือ
หนังสืออาร์เรย์จากbookReducer ผูกทั้งหมดนี้เข้าด้วยกัน เข้าถึงอาร์เรย์หนังสือและแผนที่
เหนือหนังสือเพื่อแสดงใน UI ดังที่แสดงในรายการต่อไปนี้ (Books.js)
ฟังก์ชั่นที่ให้

วัตถุสถานะ Redux ทั่วโลก
คืนสินค้า
วัตถุ
จากนี้
การทำงาน
ผ่านในหนังสือ
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 114


```
นำเข้าปฏิกิริยาจาก 'ปฏิกิริยา'
นำเข้า {
ข้อความ,
ดู,
เลื่อนดู,
สไตล์ชีต
} จาก 'react-native'
นำเข้า { เชื่อมต่อ } จาก 'react-redux'
หนังสือเรียนขยาย React.Component<{}> {
เรนเดอร์ () {
const { หนังสือ } = this.props
กลับ (
<ดู style={styles.container}>
<Text style={styles.title}>หนังสือ</Text>
<ScrollView
keyboardShouldPersistTaps='เสมอ'
style={styles.booksContainer}
>
{
books.map((หนังสือ, ดัชนี) => (
<ดู style={styles.book} คีย์={index}>
<Text style={styles.name}>{book.name}</Text>
<Text style={styles.author}>{book.author}</Text>
</ดู>
))
}
</ScrollView>
```

</ดู>
)
}
}
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์: {
ดิ้น: 1
},
คอนเทนเนอร์หนังสือ: {
ขอบด้านบนกว้าง: 1,
borderTopColor: '#ddd',
ดิ้น: 1
},
ชื่อ: {
paddingTop: 30,
paddingด้านล่าง: 20,
ขนาดตัวอักษร: 20,
textAlign: 'ศูนย์'
},
หนังสือ: {
ช่องว่างภายใน: 20
},
ชื่อ: {
นำเข้าเชื่อมต่อจาก react-redux
เนื่องจากมีการส่งคืนอาร์เรย์หนังสือ
จากฟังก์ชั่นการเชื่อมต่อ (ที่ด้านล่าง
ของรายการรหัส) คุณสามารถเข้าถึงได้
เป็นอุปกรณ์ประกอบฉาก
แผนที่เหนืออาร์เรย์ กำลังแสดง
ชื่อและผู้แต่งหนังสือแต่ละเล่ม
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 115

187
การเพิ่มการกระทำ
ขนาดตัวอักษร: 18
},
ผู้เขียน: {
ขนาดตัวอักษร: 14,
สี: '#999'

```
}
})
const mapStateToProps = (สถานะ) => ({
หนังสือ: state.bookReducer.books
})
ส่งออกการเชื่อมต่อเริ่มต้น (mapStateToProps) (หนังสือ)
```

คุณเริ่มต้นด้วยการนำเข้าการเชื่อมต่อจาก react-redux ในรายการ 8.5 คุณเขียน func-
การส่งคืนอุปกรณ์ประกอบฉากแบบอินไลน์ รายการนี้แยกและชื่อมันmapStateToProps ,
ตามแบบแผนของระบบนิเวศ Redux แบบแผนการตั้งชื่อนี้ทำมาก
มีเหตุผลเพราะคุณกำลังแมปสถานะ Redux กับอุปกรณ์ประกอบฉาก นี้
ฟังก์ชันรับสถานะ Redux เป็นอาร์กิวเมนต์และส่งกลับวัตถุที่มีหนึ่งคีย์ con-
ในประเด็นหนังสืออาร์เรย์จากbookReducer สุดท้าย คุณส่งออกฟังก์ชันการเชื่อมต่อ
ส่งผ่านmapStateToPropsเป็นอาร์กิวเมนต์แรกในการเชื่อมต่อและBooksเป็นอาร์กิวเมนต์เท่านั้น
ข้อโต้แย้งในชุดที่สองของการขัดแย้งในการเชื่อมต่อ
หลังจากเปิดแอปพลิเคชั่น คุณจะเห็นรายการหนังสือพื้นฐานดังที่แสดงไว้ก่อนหน้านี้
ในรูปที่ 8.3

8.7 การเพิ่มการกระทำ

เมื่อคุณเข้าถึงสถานะ Redux ได้แล้ว ขั้นตอนต่อไปที่สมเหตุสมผลคือการเพิ่มฟังก์ชันบางอย่าง-
ที่จะช่วยให้คุณเพิ่มหนังสือในอาร์เรย์หนังสือ Redux store เมื่อต้องการทำเช่นนี้ คุณจะใช้
การกระทำ การกระทำนั้นเป็นฟังก์ชันที่ส่งคืนวัตถุที่ส่งข้อมูลไปยังร้านค้าและ
ปรับปรุงตัวลด; เป็นวิธีเดียวที่จะเปลี่ยนร้าน แต่ละการกระทำควรมี a
พิมพ์คุณสมบัติเพื่อให้ตัวลดสามารถใช้งานได้ นี่คือตัวอย่างบางส่วน
ของการกระทำ:

```
ฟังก์ชั่น fetchBooks () {
กลับ {
ประเภท: 'FETCH_BOOKS'
}
}
ฟังก์ชั่น addBook (หนังสือ) {
กลับ {
ประเภท: 'Add_BOOK',
หนังสือ: หนังสือ
}
}
```

การดำเนินการเมื่อถูกเรียกโดยใช้ฟังก์ชันการจัดส่ง Redux จะถูกส่งไปยังตัวลดทั้งหมดใน
แอปพลิเคชันเป็นอาร์กิวเมนต์ที่สองของตัวลด (เราจะอธิบายวิธีการแนบ Redux
ฟังก์ชั่นการจัดส่งในบทนี้) เมื่อตัวลดได้รับการดำเนินการ คุณ
รับสถานะ Redux และส่งคืนอ็อบเจ็กต์

ด้วยกุญแจที่บรรจุอาเรย์หนังสือ
ส่งออกฟังก์ชันการเชื่อมต่อ
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 116

ตรวจสอบคุณสมบัติของประเภทการกระทำและอัปเดตสิ่งที่ตัวลดส่งคืนโดยพิจารณาจากว่า
การกระทำเป็นสิ่งที่กำลังฟังอยู่
ในกรณีนี้ การดำเนินการเดียวที่คุณต้องการสำหรับขั้นตอนต่อไปคือaddBookเพื่อเพิ่มเพิ่มเติม
หนังสือไปยังอาร์เรย์ของหนังสือ ใน actions.js ให้สร้างการกระทำต่อไปนี้

```
ส่งออก const ADD_BOOK = 'ADD_BOOK'
ฟังก์ชั่นการส่งออก addBook (หนังสือ) {
กลับ {
ชนิด: ADD_BOOK,
หนังสือ
}
}
```

ถัดไป วางสายbookReducerเพื่อใช้การดำเนินการaddBook

```
นำเข้า { ADD_BOOK } จาก '../actions'
const initialState = {
หนังสือ: [{ ชื่อ: 'East of Eden' ผู้แต่ง: 'John Steinbeck' }]
}
const bookReducer = (state = initialState, การกระทำ) => {
สวิตช์ (action.type) {
กรณี ADD_BOOK:
กลับ {
หนังสือ: [
...state.books,
action.book
]
}
ค่าเริ่มต้น:
คืนสถานะ
}
}
ส่งออก bookReducer เริ่มต้น
```

ในรายการ หากประเภทการดำเนินการเท่ากับADD_BOOKคุณจะส่งคืนอาร์เรย์หนังสือใหม่
taining รายการก่อนหน้าทั้งหมดในอาร์เรย์ คุณทำได้โดยสร้างอาร์เรย์ใหม่โดยใช้คำสั่ง
ตัวดำเนินการกระจายเพื่อเพิ่มเนื้อหาของอาร์เรย์หนังสือที่มีอยู่ไปยังอาร์เรย์ใหม่และ
เพิ่มรายการใหม่ให้กับอาร์เรย์ซึ่งเป็นคุณสมบัติหนังสือของการดำเนินการ
นั่นคือทั้งหมดที่คุณต้องทำในการกำหนดค่า Redux เพื่อให้ทำงานได้ สุดท้าย
ขั้นตอนคือไปที่ UI และเชื่อมต่อทั้งหมดเข้าด้วยกัน ในการรับข้อมูลหนังสือของผู้ใช้ คุณต้อง
สร้างแบบฟอร์ม รูปที่ 8.4 แสดงหน้าตาของ UI
แบบฟอร์มนี้มีสองอินพุต: หนึ่งสำหรับชื่อหนังสือและอีกอันสำหรับชื่อผู้แต่ง มัน
มีปุ่มส่งด้วย เมื่อผู้ใช้พิมพ์ลงในแบบฟอร์มคุณต้องติดตาม
ค่าในสถานะท้องถิ่น จากนั้นคุณสามารถส่งต่อค่าเหล่านั้นไปยังการกระทำเมื่อ
ผู้ใช้คลิกปุ่มส่ง
สร้างและส่งออก ADD_BOOK
ค่าคงที่สำหรับการนำกลับมาใช้ใหม่ในตัวลด
สร้างฟังก์ชัน addBook
ซึ่งรับวัตถุหนังสือเล่มเดียว
และส่งคืนวัตถุที่มี
ประเภทและหนังสือผ่านเข้า
นำเข้าค่าคงที่ ADD_BOOK
จากไฟล์การกระทำ
เพิ่มวินาที
อาร์กิวเมนต์เพื่อ
ตัวลดหนังสือ:
การกระทำ
สร้างคำสั่งสวิตช์ว่า
จะเปิดประเภทการกระทำ
หากประเภทการดำเนินการเท่ากับ ADD_BOOK
ส่งคืนอาร์เรย์หนังสือใหม่
หากคำสั่ง switch ไม่
ตีกลับสถานะที่มีอยู่
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

189
การเพิ่มการกระทำ
เปิด Books.js และนำเข้าส่วนประกอบเพิ่มเติมที่จำเป็นสำหรับฟังก์ชันนี้
เช่นเดียวกับฟังก์ชันaddBookจากการดำเนินการ คุณจะต้องสร้างinitialState . ด้วย
ตัวแปรที่จะใช้เป็นสถานะองค์ประกอบภายใน

```
นำเข้าปฏิกิริยาจาก 'ปฏิกิริยา'
นำเข้า {
ข้อความ,
ดู,
เลื่อนดู,
สไตล์ชีต,
อินพุตข้อความ,
TouchableOpacity
} จาก 'react-native'
นำเข้า { addBook } จาก './actions'
นำเข้า { เชื่อมต่อ } จาก 'react-redux'
const initialState = {
ชื่อ: '',
ผู้เขียน: ''
}
...
```

รูปที่ 8.4 UI ที่มีการเพิ่มข้อความเพื่อจับภาพ
หนังสือและชื่อผู้แต่ง
นำเข้า TextInput และ TouchableOpacity
นำเข้าฟังก์ชัน addBook
จากไฟล์การกระทำ
สร้างอ็อบเจ็กต์ initialState
มีฟิลด์ชื่อและผู้เขียน
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 118

190
C HAPTER 8 การใช้ไลบรารีสถาปัตยกรรมข้อมูล Redux
ถัดไป ในเนื้อหาของคลาส คุณต้องสร้างสามสิ่ง: สถานะคอมโพเนนต์ a
เมธอดที่ติดตามสถานะของส่วนประกอบเมื่อค่าtextInputเปลี่ยนไป และ a
วิธีการที่จะส่งการดำเนินการไปยัง Redux ที่มีค่าหนังสือ (ชื่อและผู้แต่ง)

เมื่อกดปุ่มส่ง ก่อนที่จะทำให้วิธีการเพิ่มรหัสต่อไป

```
หนังสือเรียนขยาย React.Component {
state = เริ่มต้น สถานะ
updateInput = (คีย์, ค่า) => {
this.setState({
...this.state,
[คีย์]: ค่า
})
}
addBook = () => {
this.props.dispatchAddBook (this.state)
this.setState (ค่าเริ่มต้น)
}
...
```

addBookวิธีการเรียกฟังก์ชั่นที่คุณมีการเข้าถึงเป็นอุปกรณ์ประกอบฉากจากการเชื่อมต่อ
ฟังก์ชั่น: dispatchAddBook ฟังก์ชันนี้ยอมรับทั้งสถานะเป็นอาร์กิวเมนต์
ซึ่งเป็นวัตถุที่มีคุณสมบัติชื่อและผู้เขียน หลังจากดำเนินการจัดส่งเรียบร้อยแล้ว
คุณล้างสถานะคอมโพเนนต์โดยรีเซ็ตเป็นค่าinitialState
ด้วยฟังก์ชันที่มีอยู่ คุณสามารถสร้าง UI และเชื่อมโยงวิธีการเหล่านี้ได้ถึง
มัน. ใต้แท็กปิดของScrollViewใน Books.js ให้เพิ่มแบบฟอร์ม UI

```
หนังสือเรียนขยาย React.Component {
...
เรนเดอร์ () {
...
</ScrollView>
<ดู style={styles.inputContainer}>
<ดู style={styles.inputWrapper}>
<ป้อนข้อความ
ค่า={this.state.name}
onChangeText={value => this.updateInput('ชื่อ', ค่า)}
style={styles.input}
ตัวยึด = 'ชื่อหนังสือ'
/>
<ป้อนข้อความ
ค่า={this.state.author}
onChangeText={value => this.updateInput('ผู้เขียน', ค่า)}
style={styles.input}
ตัวยึด = 'ชื่อผู้แต่ง'
/>
</ดู>
```

ให้สถานะส่วนประกอบ

ค่าของตัวแปร initialState
สร้างวิธีการ updateInput ที่
รับสองอาร์กิวเมนต์: คีย์และค่า
คุณจะอัปเดตสถานะโดยใช้ปุ่ม
ตัวดำเนินการกระจายเพื่อเพิ่มที่มีอยู่
ระบุคู่คีย์-ค่ากับสถานะใหม่
แล้วเพิ่มคู่คีย์-ค่าใหม่
โทรdispatchAddBook,
เข้าถึงได้จาก
ฟังก์ชั่นการเชื่อมต่อ
รับเมธอด updateInput เป็น
คุณสมบัติของ onChangeText ผ่าน 'ชื่อ'
หรือ 'ผู้เขียน' เป็นอาร์กิวเมนต์แรกและ
ค่าของ TextInput เป็นอาร์กิวเมนต์ที่สอง
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 119

191
การเพิ่มการกระทำ

```
<TouchableOpacity onPress={this.addBook}>
<ดู style={styles.addButtonContainer}>
<Text style={styles.addButton}>+</Text>
</ดู>
</TouchableOpacity>
</ดู>
</ดู>
}
}
รูปแบบ const = StyleSheet.create ({
คอนเทนเนอร์อินพุต: {
ช่องว่างภายใน: 10,
พื้นหลังสี: '#ffffff',
borderTopColor: '#ededed',
ขอบด้านบนกว้าง: 1,
flexDirection: 'แถว',
ส่วนสูง: 100
},
```

```
inputWrapper: {
ดิ้น: 1
},
ป้อนข้อมูล: {
ความสูง: 44,
ช่องว่างภายใน: 7,
backgroundColor: '#ededed',
borderColor: '#ddd',
ขอบกว้าง: 1,
รัศมีขอบ: 10,
ดิ้น: 1,
ขอบด้านล่าง: 5
},
เพิ่มปุ่ม: {
ขนาดตัวอักษร: 28,
ความสูงของเส้น: 28
},
addButtonContainer: {
ความกว้าง: 80,
ความสูง: 80,
backgroundColor: '#ededed',
ระยะขอบซ้าย: 10,
justifyContent: 'ศูนย์',
alignItems: 'ศูนย์',
borderRadius: 20
},
...
}
const mapDispatchToProps = {
dispatchAddBook: (หนังสือ) => addBook (หนังสือ)
}
ส่งออกการเชื่อมต่อเริ่มต้น (mapStateToProps, mapDispatchToProps) (หนังสือ)
}
```

เรียกเมธอด addBook TouchableOpacity

ล้อมองค์ประกอบ View ไว้ ปล่อยให้เป็น

ตอบสนองต่อการสัมผัสอย่างถูกต้อง

เพิ่มรูปแบบใหม่

สร้างวัตถุ mapDispatchToProps

ผ่านใน mapDispatchToProps เป็น

อาร์กิวเมนต์ที่สองในการเชื่อมต่อ

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc

facebook.com/somsacki

หน้า 120

192

C HAPTER 8 การใช้ไลบรารีสถาปัตยกรรมข้อมูล Redux

ในอ็อบเจ็กต์mapDispatchToPropsคุณสามารถประกาศฟังก์ชันที่คุณต้องการเข้าถึงเป็น
อุปกรณ์ประกอบฉากในองค์ประกอบ คุณสร้างฟังก์ชันใหม่ที่เรียกว่าdispatchAddBookและ have
มันเรียกการกระทำaddBookส่งผ่านหนังสือเป็นอาร์กิวเมนต์ คล้ายกับที่mapStateTo-
อุปกรณ์ประกอบฉากจะแมปสถานะกับอุปกรณ์ประกอบฉาก, mapDispatchToProps จะจับคู่การกระทำ (ที่จำเป็น
เพื่อส่งไปยังเครื่องลดขนาด) ไปยังอุปกรณ์ประกอบฉาก เพื่อให้การดำเนินการได้รับการยอมรับ-
nized โดย Redux reducers จะต้องประกาศในวัตถุmapDispatchToPropsนี้
คุณส่งผ่านmapDispatchToPropsเป็นอาร์กิวเมนต์ที่สองไปยังฟังก์ชันการเชื่อมต่อ
ตอนนี้ คุณควรจะสามารถเพิ่มหนังสือในรายการหนังสือได้อย่างง่ายดาย

8.8 การลบรายการจากร้าน Redux ในตัวลดขนาด

ขั้นตอนต่อไปคือการเพิ่มวิธีการลบหนังสือที่คุณอ่านแล้ว ให้ทุก-
สิ่งที่คุณได้รวบรวมมา นี้จะไม่ต้องทำงานมากเกินไป (รูปที่ 8.5)
สิ่งแรกที่ต้องนึกถึงเมื่อลบรายการออกจากอาร์เรย์เช่นนี้คือ
วิธีการระบุหนังสือว่ามีเอกลักษณ์เฉพาะตัว ตอนนี้ ผู้ใช้สามารถมีหนังสือได้หลายเล่ม
กับผู้แต่งคนเดียวกันหรือหนังสือหลายเล่มที่มีชื่อเดียวกัน ดังนั้นโดยใช้พร็อพ-
erties จะไม่ทำงาน คุณสามารถใช้ไลบรารีเช่น uuid เพื่อสร้างตัวระบุเฉพาะแทนได้
ในขณะที่บิน. เพื่อเริ่มการตั้งค่านี้ จากบรรทัดคำสั่ง ติดตั้งไลบรารี uuid ลงใน
node_modules :

npm ฉัน uuid --save

รูปที่ 8.5 การเพิ่มปุ่มลบไปยัง
Books.js UI

ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 121

193

การลบรายการจากร้าน Redux ในตัวลดขนาด

ถัดไป คุณจะใช้ตัวระบุเฉพาะในตัวลดสำหรับรายการใน
initialState หนังสืออาร์เรย์ ใน reducers/bookReducer.js ให้อัปเดตการนำเข้า
และinitialStateให้ดูเหมือนรายการถัดไป

นำเข้า uuidV4 จาก 'uuid/v4'

นำเข้า { ADD_BOOK } จาก '../actions'
const initialState = {
หนังสือ: [{ ชื่อ: 'East of Eden' ผู้แต่ง: 'John Steinbeck', id: uuidV4() }]
}
ไลบรารี uuid มีอัลกอริธึมไม่กี่แบบให้เลือก ที่นี่ คุณนำเข้าเฉพาะ v4 algo-
ritm ซึ่งสร้างสตริงอักขระ 32 ตัวแบบสุ่ม จากนั้นคุณเพิ่มคุณสมบัติใหม่ให้กับ
initialState หนังสืออาร์เรย์ ID และสร้างตัวระบุที่ไม่ซ้ำกันใหม่โดยการเรียก uuidV4 ()
ตอนนี้คุณมีวิธีระบุรายการต่างๆ ในอาร์เรย์หนังสือแล้ว คุณ
พร้อมก้าวไปข้างหน้ากับฟังก์ชั่นที่เหลือ ขั้นตอนต่อไปคือการสร้าง
การกระทำใหม่ใน actions.js; คุณจะเรียกมันว่าเมื่อคุณต้องการนำหนังสือออก คุณยังต้อง
อัปเดตการดำเนินการaddBookเพื่อเพิ่ม ID ให้กับหนังสือที่สร้างขึ้นใหม่

ส่งออก const ADD_BOOK = 'ADD_BOOK'
ส่งออก const REMOVE_BOOK = 'REMOVE_BOOK'
นำเข้า uuidV4 จาก 'uuid/v4'
ฟังก์ชั่นการส่งออก addBook (หนังสือ) {
กลับ {
ชนิด: ADD_BOOK,
หนังสือ: {
...หนังสือ,
id: uuidV4()
}
}
}
ฟังก์ชั่นการส่งออก removeBook (หนังสือ) {
กลับ {
ประเภท: REMOVE_BOOK,
หนังสือ
}
}
ถัดไป ตัวลดจะต้องตระหนักถึงการกระทำใหม่ ใน reducers/bookReducer.js
สร้าง listener ชนิดใหม่ อันนี้สำหรับREMOVE_BOOKและเพิ่ม functional-
ity เพื่อลบหนังสือออกจากอาร์เรย์ของหนังสือที่เก็บไว้ในสถานะ Redux
นำเข้า
อัลกอริทึม v4
เพิ่มคุณสมบัติ id ให้กับ initialState และ
สร้างตัวระบุที่ไม่ซ้ำกันใหม่
สร้างค่าคงที่ที่ใช้ซ้ำได้ REMOVE_
BOOK ใช้ที่นี่และในตัวลด

นำเข้าไลบรารี uuid
เพิ่มคีย์ใหม่ให้กับหนังสือ กำหนด
id คุณสมบัติของเอกลักษณ์ที่สร้างขึ้นใหม่
ตัวระบุโดยใช้ฟังก์ชัน uuidV4
สร้างฟังก์ชัน removeBook ใหม่
ที่ส่งคืนวัตถุที่มีประเภทและ
พารามิเตอร์หนังสือที่ส่งผ่านใน
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

หน้า 122


```
นำเข้า uuidV4 จาก 'uuid/v4'
นำเข้า { ADD_BOOK, REMOVE_BOOK } จาก '../actions'
const initialState = {
หนังสือ: [{ ชื่อ: 'East of Eden' ผู้แต่ง: 'John Steinbeck', id: uuidV4() }]
}
const bookReducer = (state = initialState, การกระทำ) => {
สวิตช์ (action.type) {
...
กรณี REMOVE_BOOK:
ดัชนี const = state.books.findIndex (
book => book.id === action.book.id)
กลับ {
หนังสือ: [
...state.books.slice(0, ดัชนี),
...state.books.slice(ดัชนี + 1)
]
}
...
}
}
ส่งออก bookReducer เริ่มต้น
```

สิ่งสุดท้ายที่ต้องทำคือใช้ฟังก์ชันremoveBookใหม่นี้ใน UI ของ
ส่วนประกอบหนังสือ(Books.js) คุณจะนำเข้าการดำเนินการ removeBook เพิ่มการลบ แต่-
ตันกับแต่ละรายการที่แสดงผล และเชื่อมโยงปุ่มเอาออกกับการดำเนินการremoveBook

```
...
นำเข้า { addBook, removeBook } จาก './actions'
...
removeBook = (หนังสือ) => {
this.props.dispatchRemoveBook (หนังสือ)
}
...
{
books.map((หนังสือ, ดัชนี) => (
<ดู style={styles.book} คีย์={index}>
<Text style={styles.name}>{book.name}</Text>
<Text style={styles.author}>{book.author}</Text>
<Text onPress={() => this.removeBook(book)}>
ลบ
</Text>
</ดู>
))
}
...
```

นำเข้า REMOVE_BOOK . ใหม่

ค่าคงที่จากโฟลเดอร์การกระทำ

เพิ่มกรณีใหม่ให้กับคำสั่งสวิตช์ว่า

ฟังสำหรับประเภทการดำเนินการ REMOVE_BOOK

ค้นหาดัชนีของหนังสือที่จะลบ

ส่งคืน a

อาร์เรย์ใหม่

ประกอบด้วย

ครั้งแรกและ

ครึ่งหลัง

ของ

ที่มีอยู่

อาร์เรย์หนังสือ,

ออกไป

ดัชนีของ

หนังสือถึง

ถูกลบออก

เพิ่ม removeBook เป็นการนำเข้า

จากไฟล์การกระทำ

สร้างวิธีการเรียนใหม่ removeBook

เรียก this.props.dispatchRemoveBook
เป็นคีย์ใหม่ใน mapDispatchToProps
ส่งกลับองค์ประกอบข้อความใหม่และ
แนบ removeBook เข้ากับเหตุการณ์ onPress
ค้นหา Hybrid Mobile App https://tinyurl.com/HybridApp-BSc
facebook.com/somsacki

195
สรุป

```
const mapDispatchToProps = {
dispatchAddBook: (หนังสือ) => addBook (หนังสือ),
dispatchRemoveBook: (หนังสือ) => removeBook (หนังสือ)
}
...
```

สรุป

- ด้วยบริบท คุณสามารถส่งคุณสมบัติและข้อมูลไปยังลูกใน React Native
แอปพลิเคชันโดยไม่ส่งต่อคุณสมบัติไปยังลูกแต่ละคนอย่างชัดเจน
- ตัวลดจะคล้ายกับที่เก็บข้อมูลแบบดั้งเดิมในแง่ที่พวกมันติดตาม
และส่งคืนข้อมูล แต่ยังอนุญาตให้คุณอัปเดตข้อมูลในร้านค้า
- คุณสามารถสร้างและใช้การดำเนินการเพื่ออัปเดตร้าน Redux
- ด้วยฟังก์ชันการเชื่อมต่อคุณสามารถเข้าถึงข้อมูลจากสถานะ Redux เป็นอุปกรณ์ประกอบฉาก
และสร้างฟังก์ชันการจัดส่งที่โต้ตอบกับตัวลดขนาดโดยใช้การกระทำ
- เมื่อใดก็ตามที่จำเป็นต้องเปลี่ยนข้อมูลในตัวลด จะต้องดำเนินการโดยใช้การกระทำ